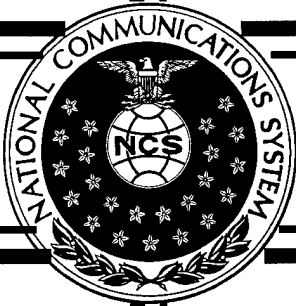


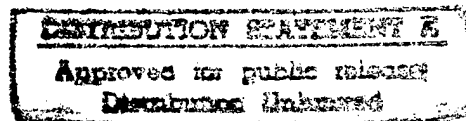
NCS TIB 95-2



# NATIONAL COMMUNICATIONS SYSTEM

TECHNICAL INFORMATION BULLETIN 95-2

COLOR FACSIMILE



FEBRUARY 1995

OFFICE OF THE MANAGER  
NATIONAL COMMUNICATIONS SYSTEM  
701 SOUTH COURT HOUSE ROAD  
ARLINGTON, VA 22204-2198

19970117 159

ALL QUALITY INFORMATION 1

NFS-4758

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE February 1995		3. REPORT TYPE AND DATES COVERED Final Report
4. TITLE AND SUBTITLE  Color Facsimile			5. FUNDING NUMBERS  DCA100-91-C-0031	
6. AUTHOR(S)  Stephen Perschau				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Delta Information Systems, Inc. Bldg 3, Suite 120 300 Welsh Road Horsham, PA 19044-2273			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Communications System Office of Technology and Standards Division 701 South Court House Road Arlington, Virginia 22204-2198			10. SPONSORING/MONITORING AGENCY REPORT NUMBER  NCS TIB #95-2	
11. SUPPLEMENTARY NOTES  This report supersedes NCS TIB #93-17.				
12a. DISTRIBUTION/AVAILABILITY STATEMENT  Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) The purpose of this project was to continue the color facsimile work started under a previous task, including the evaluation of the use of default Huffman tables, optimized ("Custom") Huffman tables, default quantization tables, and scaled quantization tables. Included in this effort was the modification of existing JPEG compression and decompression software available from Independent JPEG Users Group to process CIELAB color images and to use externally specified Huffman tables. In addition a conversion program was written to convert CIELAB color space images to red, green, blue color space images to allow viewing of the images by a commercially available viewing package such as HIJACK. This report provides a brief description of the objectives of the task, provides some background of the use of custom vs. default quantization matrices and Huffman coding tables for transmitting color FAX images by the JPEG baseline standard, includes the software development required in order to support the evaluation runs along with the results of the evaluation runs themselves and the conclusions that can be drawn from them, discusses comparing Huffman vs Informational coding, and discusses possible future plans.				
14. SUBJECT TERMS Color Imagery Color Facsimile			15. NUMBER OF PAGES 60	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT  UNCLASS	18. SECURITY CLASSIFICATION OF THIS PAGE  UNCLASS	19. SECURITY CLASSIFICATION OF ABSTRACT  UNCLASS	20. LIMITATION OF ABSTRACT  UNLIMITED	

## GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to *stay within the lines* to meet *optical scanning requirements*.

**Block 1. Agency Use Only (Leave blank).**

**Block 2. Report Date.** Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

**Block 3. Type of Report and Dates Covered.** State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

**Block 4. Title and Subtitle.** A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

**Block 5. Funding Numbers.** To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

<b>C</b> - Contract	<b>PR</b> - Project
<b>G</b> - Grant	<b>TA</b> - Task
<b>PE</b> - Program Element	<b>WU</b> - Work Unit Accession No.

**Block 6. Author(s).** Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

**Block 7. Performing Organization Name(s) and Address(es).** Self-explanatory.

**Block 8. Performing Organization Report Number.** Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

**Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es).** Self-explanatory.

**Block 10. Sponsoring/Monitoring Agency Report Number.** (If known)

**Block 11. Supplementary Notes.** Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

**Block 12a. Distribution/Availability Statement.** Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

**DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."

**DOE** - See authorities.

**NASA** - See Handbook NHB 2200.2.

**NTIS** - Leave blank.

**Block 12b. Distribution Code.**

**DOD** - Leave blank.

**DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.

**NASA** - Leave blank.

**NTIS** - Leave blank.

**Block 13. Abstract.** Include a brief (*Maximum 200 words*) factual summary of the most significant information contained in the report.

**Block 14. Subject Terms.** Keywords or phrases identifying major subjects in the report.

**Block 15. Number of Pages.** Enter the total number of pages.

**Block 16. Price Code.** Enter appropriate price code (*NTIS only*).

**Blocks 17. - 19. Security Classifications.** Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

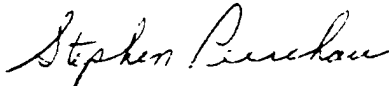
**Block 20. Limitation of Abstract.** This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

NCS TECHNICAL INFORMATION BULLETIN 95-2

COLOR FACSIMILE

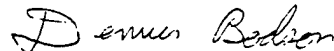
FEBRUARY 1995

PROJECT OFFICER



STEPHEN PERSCHAU  
Computer Scientist  
Office of Technology  
and Standards

APPROVED FOR PUBLICATION:



DENNIS BODSON  
Assistant Manager  
Office of Technology  
and Standards

FOREWORD

Among the responsibilities assigned to the Office of the Manager, National Communications System, is the management of the Federal Telecommunication Standards Program. Under this program, the NCS, with the assistance of the Federal Telecommunication Standards Committee identifies, develops, and coordinates proposed Federal Standards which either contribute to the interoperability of functionally similar Federal telecommunication systems or to the achievement of a compatible and efficient interface between computer and telecommunication systems. In developing and coordinating these standards, a considerable amount of effort is expended in initiating and pursuing joint standards development efforts with appropriate technical committees of the International Organization for Standardization, and the International Telegraph and Telephone Consultative Committee of the International Telecommunication Union. This Technical Information Bulletin presents an overview of an effort which is contributing to the development of compatible Federal, national, and international standards in the area of facsimile. It has been prepared to inform interested Federal activities of the progress of these efforts. Any comments, inputs or statements of requirements which could assist in the advancement of this work are welcome and should be addressed to:

Office of the Manager  
National Communications System  
Attn: NT  
701 S. Court House Road  
Arlington, VA 22204-2198

**TASK 2  
TECHNICAL WORK IN THE  
AREA OF FACSIMILE**

**SUBTASK 3  
COLOR FACSIMILE**

**FINAL REPORT  
CONTRACT DCA100-91-C-0031  
OPTION YEAR 3**

**February, 1995**

**Submitted to:  
NATIONAL COMMUNICATIONS SYSTEM  
ARLINGTON, VA**

**DELTA INFORMATION SYSTEMS, INC.  
300 Welsh Road, Ste. 120  
Horsham, PA 19044-2273**

**TEL: (215) 657-5270**

**FAX: (215) 657-5273**

## TABLE OF CONTENTS

1.0 INTRODUCTION .....	1 - 1
2.0 BACKGROUND .....	2 - 1
2.1 The Elements of JPEG .....	2 - 1
2.2 Default Huffman Coding Tables .....	2 - 1
3.0 SUMMARY OF WORK PERFORMED .....	3 - 1
3.1 Software Modifications .....	3 - 1
3.1.1 CIELAB Image Conversion .....	3 - 1
3.1.2 JPEG Software Modifications .....	3 - 3
3.1.3 CIELAB to RGB Color Space Conversion .....	3 - 3
3.2 Results of Evaluation Runs .....	3 - 4
3.3 Discussion of Results .....	3 - 8
3.4 Conclusions .....	3 - 9
4.0 HUFFMAN VS. INFORMATIONAL CODING .....	4 - 1
4.1 Introduction .....	4 - 1
4.2 Alphabets .....	4 - 1
4.3 Analysis .....	4 - 2
4.4 The Comparison .....	4 - 4
4.5 Experiments .....	4 - 4
4.6 Results .....	4 - 5
4.7 Observations and Discussion .....	4 - 6
4.8 Conclusion .....	4 - 7
5.0 FUTURE PLANS .....	5 - 1
5.1 Determining Optimal Default Huffman Code .....	5 - 1
5.1.1 Criterion for Determining Required Sample Sizes .....	5 - 1
5.1.2 Approach to Best Default Huffman Codes .....	5 - 2

## APPENDICES

A - Image Histograms

B - Contribution D10 - Study Group 8

C - Black and White Copies of Color Images Used for Evaluations

## 1.0 INTRODUCTION

This document summarizes work performed by Delta Information Systems, Inc. (DIS) for the National Communications System (NCS), Office of Technology and Standards. This office is responsible for the management of the Federal Telecommunications Standards Program, which develops telecommunications standards, whose use is mandatory for all Federal departments and agencies. The purpose of this project, performed under Task 2, Subtask 3 of contract number DCA100-91-C-0031 during Option Year 3, was to continue the work on color facsimile that was begun under a previous task.

The digital transmission of color imagery is of particular importance to the Government for transmission of photographs, half tones, maps and fingerprints. The ITU-T is now in the process of developing standards for the transmission of color imagery as part of the facsimile recommendations, including both Group 3 facsimile and Group 4 facsimile.

The purpose of this project was to continue the color facsimile work started under a previous task, including the evaluation of the use of default Huffman tables, optimized ("Custom") Huffman tables, default quantization tables, and scaled quantization tables. Included in this effort was the modification of existing Joint Photographic Experts Group (JPEG) compression and decompression software available from the Independent JPEG Users Group to process Commission Internationale de l'Eclairage  $L^*a^*b^*$  (CIELAB) color images and to use externally specified Huffman tables. In addition a conversion program was written to convert CIELAB color space images to red, green, blue (RGB) color space images to allow viewing of the images by a commercially available viewing package such as HIJACK.

This report is comprised of five sections. Section 1.0 provides a brief description of the objectives of the task and an outline of the contents of this report.

Section 2.0 provides some background on the use of custom vs. default quantization matrices and Huffman coding tables for transmitting color FAX images by the JPEG (Joint Photographic Experts Group) baseline standard.

Section 3.0 is a summary of the work performed. This includes the software development required in order to support the evaluation runs along with the results of the evaluation runs themselves and the conclusions that can be drawn from them.

Section 4.0 is a technical discussion comparing Huffman vs Informational coding.

Section 5.0 is a discussion of possible future plans.

## 2.0 BACKGROUND

### 2.1 The Elements of JPEG

When one peels away all the detailed specifications of the JPEG system and examines its basic ingredients, one finds the three essential elements of a good data compression system for continuous-tone imagery: (1) a data compactor (the DCT), which packs most of a block of data into a few frequency coefficients, (2) a quantizer (the quantization matrices and compression scale factor), which controls the trade-off between data compression and fidelity, and (3) a variable length or "entropy" coder (Huffman or arithmetic coder) to encode the data that must be transmitted after compaction and quantization. Most recent work has been aimed at employing JPEG for some specific purpose, such as color facsimile, and fine tuning the quantization matrices and Huffman codes. This report describes the work done by Delta Information Systems using various Huffman Tables (JPEG default, optimized etc.) within the JPEG compression algorithm.

### 2.2 Default Huffman Coding Tables

One of the outstanding issues of the past few years has been whether there is a good default Huffman code. A "custom" Huffman code, optimized for the image to be transmitted, will always perform at least as well as any other Huffman code, and usually better. Normally the use of a custom code has two major drawbacks. One is that the transmitter must make two coding passes: one to collect the image statistics and build the codes, and one to encode and transmit the data. The other drawback is that the transmitter must transmit the custom coding tables to the receiver. It should be noted that the current standard for Group 3 facsimile requires the sending of the Huffman tables independent of whether they are custom or not. Because of this operational requirement, only the first drawback exists for Group 3 facsimile implementations.

Much of the effort devoted to the current study has consisted of going back to the basics of information theory to show that, in principle, it is straightforward to build a default Huffman code (or a separate code for each of the various transmission parameters and image classes) that will perform as well as or better than any other *fixed* (image independent) Huffman code. One of the basic tenets of information theory is the notion of *entropy*, based upon the probability function of the symbols being encoded. The entropy is the theoretical lower bound on the long-term average code word length per symbol for that probability function when the symbols are coded independently. A Huffman code derived from the same probability function is guaranteed to produce a long-term average bit rate that is no more than one bit per symbol greater than the entropy, and often exceeds the entropy by a much smaller amount than one bit per symbol. Moreover, this Huffman code is optimal in the sense that no fixed Huffman code can yield a smaller average number of bits per coded symbol.



The main practical impediment to generating this optimal fixed Huffman code is the requirement that a very large number of symbols be sampled to yield a reliable estimate of the probability function. Section 3 describes experiments in which various Huffman codes were evaluated. The main conclusion drawn from the results of these experiments was that the data sample sizes were insufficient to yield close estimates of the probability function. If a number of independent experimenters were to collect a sufficient number of random samples from which to estimate the probability function, then the law of large numbers says that the estimated probability functions, and hence the Huffman codes, would be very close to one another.

### **3.0 SUMMARY OF WORK PERFORMED**

This section details the work performed by Delta Information Systems in evaluating the use of various Huffman tables in the JPEG compression algorithm. This work was divided into three tasks. The first task was the software effort required to support the evaluation runs. The second task was the actual processing of the evaluation runs themselves. The third and final task was the review of the results of the runs and the formation of conclusions. The details of each of these tasks are discussed in the next three sections.

#### **3.1 Software Modifications**

In order to perform the Huffman table evaluation runs, software was needed to be written and/or modified for the following functions:

- Conversion of the raw CIELAB color images into a file format suitable for processing.
- Modifications to the JPEG users group compression and decompression software to process CIELAB images and to allow the use of externally specified Huffman tables.
- Software to convert a CIELAB color image to a Targa format RGB color image file for viewing.

##### **3.1.1 CIELAB Image Conversion**

The CIELAB color image files acquired for the JPEG Huffman table evaluations were of two different formats and could not be processed directly by the JPEG users group software. One set of images contained the three color components of the image in the same file, plane interleaved. The second set of images contained the three color components of the image in three separate files. To solve this problem a Delta Information System CIELAB color image file format was defined and conversion programs were written to convert the CIELAB color images to this format prior to processing by the JPEG software. The DIS image file format consists of a header section containing the image size, data precision etc. and a detail section containing the pixel interleaved LAB color components. Shown in Figure 3.1 is the image file structure.

## Header

Image id	-	one byte set to an ASCII "L"
Image Type	-	one byte - currently unused
Image index	-	two bytes - currently unused
X origin	-	two bytes - currently set to 0 indicating left to right scan
Y origin	-	two bytes - currently set to 0 indicating top to bottom scan
Width	-	two bytes - number of pixels in a line
Length	-	two bytes - number of lines in the image
Pix depth	-	one byte - number of bits per pixel - currently 24
Image Desc	-	one byte - currently unused

## Detail

L component - pixel one line one	-	one byte
A component - pixel one line one	-	one byte
B component - pixel one line one	-	one byte
L component - pixel two line one	-	one byte
A component - pixel two line one	-	one byte
B component - pixel two line one	-	one byte

L component - pixel 'x' line one	-	one byte
A component - pixel 'x' line one	-	one byte
B component - pixel 'x' line one	-	one byte

L component - pixel 'x' line 'y'	-	one byte
A component - pixel 'x' line 'y'	-	one byte
B component - pixel 'x' line 'y'	-	one byte

**FIGURE 3.1 DIS CIELAB FILE FORMAT**

### **3.1.2 JPEG Software Modifications**

The compression and decompression software as supplied by the JPEG users group could only process the following file formats:

PPM - PBMPLUS color format  
PGM - PBMPLUS gray-scale format  
GIF  
TARGA  
RLE - Utah Raster Toolkit

Since none of these file formats was compatible with our pixel interleaved CIELAB file format, the JPEG compression and decompression software was modified to process/generate the DIS CIELAB image file format discussed in Section 3.1.1.

Another requirement of the JPEG compression software was the ability to process a color image using an externally specified Huffman table and also to save the Huffman table used for a given compression run. After evaluating the changes required to allow the manipulation of the Huffman tables within the JPEG software, it was found that it would be easier to manipulate the symbol count tables used to generate the Huffman tables. The JPEG software was therefore modified to allow the user to specify an external symbol count/histogram file. The JPEG software could then read this file and generate a Huffman table based on its contents. Additionally the JPEG compression software was modified to save the current symbol count/histogram table to a disk file for subsequent use in other evaluation runs.

### **3.1.3 CIELAB to RGB Color Space Conversion**

To view the CIELAB color image files with the commercially available software package HIJACK, software was written to convert CIELAB color images to RGB color images. The conversion of a CIELAB image file is a multiple step process consisting of the following:

- Descale the eight bit LAB components to original LAB values
- Convert the LAB color space components to the XYZ color space components
- Convert the XYZ color space components to the RGB color space components
- Scale the RGB values to eight bits for use by the viewing program

The descaling of the LAB components is image dependent but the remainder of the conversion is done according to the following equations.

The LAB to XYZ color space conversion is done using the following reverse transformations of the XYZ to LAB color space conversion.

$$\begin{aligned}
X &= X_n(((L + 16)/116 + a/500)\text{third power}) \\
Y &= Y_n(((L + 16)/116)\text{third power}) \\
Z &= Z_n(((L + 16)/116 - b/200)\text{third power})
\end{aligned}$$

where  $X_n$ ,  $Y_n$ ,  $Z_n$  are the Tristimulus values for the Reference White for a specific illuminant.

The XYZ to RGB color space conversion is done using the following inverse transformations of the RGB to XYZ color space conversion.

$$\begin{aligned}
r \text{ value} &= (1.911 * X) - (0.534 * Y) - (0.290 * Z) \\
g \text{ value} &= -(0.985 * X) + (1.999 * Y) - (0.028 * Z) \\
b \text{ value} &= (0.058 * X) - (0.119 * Y) + (0.902 * Z)
\end{aligned}$$

The resultant RGB color space values are then gamma corrected and scaled to eight bit integers for the viewing program.

### 3.2 Results of Evaluation Runs

Listed below are the results of the Huffman Table evaluation runs. The evaluation runs results are divided into two groupings. In the first group each LAB color image was compressed using the following Huffman tables:

- Optimized to the image itself
- T.81 - JPEG default Huffman table
- Huffman table from ITU-T Delayed Contribution D10 from Japan
- Delta Composite

The Delta composite Huffman table was generated from the combined histograms of the eight test images. Included in Appendix A are plots of the histograms of the four Huffman coded symbol sets used in the generation of the Delta composite Huffman table. Also included in Appendix A are the plots of the histograms of the FAXBALLS image which show how widely image histograms can differ with little negative effect on the data compression.

Although it was not part of the original scope of work, the Huffman tables were included from Contribution D10 from Japan. This was done primarily to compare the results of their Huffman table against the JPEG default and also the Delta composite. It should be noted that their "composite" Huffman table was generated using two subsamplings and only three images. For details on Contribution D10 see Appendix B.

In the second group of evaluation runs each image was compressed using the Huffman table generated by the optimized runs for each of the other LAB color images. The evaluation runs were performed at scale factors of 9, 24 and 71. The scale factor of 9 will generate a high quality image with reduced compression. The scale factor of 24 will generate an image of reasonable quality with increased

compression. The scale factor of 71 will generate a very small file of low image quality. Listed below are the results of all evaluation runs. All image file sizes are given in bytes. Black and white versions of the images processed can be found in Appendix C.

**Image SA001\_NL - Size = 12,582,926**

Scale Factor	9	24	71
Optimized	1,413,382	730,832	319,345
T.81 - JPEG	1,437,991	745,805	358,701
Contrib D10	1,431,632	741,669	353,769
Delta	1,456,336	737,763	339,987
PM003_NL	1,453,547	753,219	351,563
C1LAB	1,423,506	740,357	360,143
APPOTLAB	1,509,188	748,591	340,601
TOYSLAB	1,515,136	738,269	338,825
FAXBALLS	1,516,839	761,738	322,104
LATOUR1	1,573,038	758,688	321,980
LATOUR2	1,569,391	758,875	322,104

**Image PM003\_NL - Size = 12,582,926**

Scale Factor	9	24	71
Optimized	1,135,193	560,700	284,822
T.81 - JPEG	1,161,256	581,991	324,281
Contrib D10	1,151,224	568,895	313,302
Delta	1,177,348	572,712	305,290
SA001_NL	1,247,329	606,375	332,819
C1LAB	1,165,696	581,162	326,804
APPOTLAB	1,221,377	592,422	310,714
TOYSLAB	1,260,243	608,081	323,425
FAXBALLS	1,220,607	580,365	298,826
LATOUR1	1,281,776	600,438	297,786
LATOUR2	1,270,637	598,000	297,431

**Image C1\_LAB - Size = 15,728,658**

Scale Factor	9	24	71
Optimized	2,959,458	1,647,938	834,134
T.81 - JPEG	3,030,400	1,659,528	852,469
Contrib D10	3,050,316	1,656,829	846,693

Delta	3,138,471	1,681,105	844,083
PM003_NL	3,109,657	1,673,421	849,160
C1LAB	3,258,119	1,700,042	849,147
APPOTLAB	3,258,500	1,730,925	857,116
TOYSLAB	3,343,333	1,745,020	858,053
FAXBALLS	3,252,919	1,739,885	860,634
LATOUR1	3,474,274	1,804,264	863,499
LATOUR2	3,452,565	1,800,748	865,525

**Image APPOTLAB - Size = 24,160,886**

Scale Factor	9	24	71
Optimized	2,245,465	1,066,939	396,281
T.81 - JPEG	2,293,869	1,126,514	479,767
Contrib D10	2,275,981	1,099,993	462,139
Delta	2,291,850	1,082,191	440,976
SA001_NL	2,305,114	1,116,424	493,769
PM003_NL	2,325,993	1,106,855	438,365
C1LAB	2,278,071	1,119,215	486,525
TOYSLAB	2,340,781	1,079,254	463,780
FAXBALLS	2,427,822	1,146,556	423,937
LATOUR1	2,414,091	1,092,128	407,649
LATOUR2	2,416,806	1,090,696	408,218

**Image TOYSLAB - Size = 35,558,270**

Scale Factor	9	24	71
Optimized	3,163,614	1,524,975	593,394
T.81 - JPEG	3,227,067	1,587,727	708,666
Contrib D10	3,197,180	1,555,361	686,838
Delta	3,222,968	1,538,413	654,687
SA001_NL	3,225,164	1,553,303	710,330
PM003_NL	3,266,247	1,579,982	670,189
C1LAB	3,189,545	1,572,865	715,820
APPOTLAB	3,280,633	1,532,801	656,083
FAXBALLS	3,400,209	1,605,095	634,853
LATOUR1	3,401,297	1,550,249	596,466
LATOUR2	3,403,397	1,549,424	596,834

**Image FAXBALLS - Size = 1,572,882**

Scale Factor	9	24	71
Optimized	80,897	47,554	25,653
T.81 - JPEG	83,273	50,124	30,685
Contrib D10	82,830	49,310	29,800
Delta	85,156	49,729	28,827
SA001_NL	87,759	50,919	30,790
PM003_NL	84,533	49,864	29,284
C1LAB	83,557	50,066	31,077
APPOTLAB	87,883	51,204	29,130
TOYSLAB	90,255	51,421	29,585
LATOUR1	90,035	49,860	26,334
LATOUR2	89,636	49,749	26,353

**Image LATOUR1 - Size = 16,906,674**

Scale Factor	9	24	71
Optimized	1,262,357	533,097	220,717
T.81 - JPEG	1,288,695	570,000	284,478
Contrib D10	1,275,174	557,610	272,778
Delta	1,277,126	545,219	254,756
SA001_NL	1,296,753	568,623	291,848
PM003_NL	1,295,937	557,393	256,980
C1LAB	1,278,480	569,794	289,699
APPOTLAB	1,299,961	545,853	254,561
TOYSLAB	1,305,164	550,099	268,690
LATOUR2	1,334,552	533,601	226,526

**IMAGE LATOUR2 - Size = 16,906,674**

Scale Factor	9	24	71
Optimized	1,524,934	674,848	267,031
T.81 - JPEG	1,554,130	701,705	325,141
Contrib D10	1,538,951	688,199	313,384
Delta	1,551,592	678,911	297,435



SA001_NL	1,586,426	697,241	329,151
PM003_NL	1,564,440	694,127	302,283
C1LAB	1,537,945	698,744	329,408
APPOTLAB	1,608,170	682,037	297,007
TOYSLAB	1,617,216	684,058	306,985
FAXBALLS	1,636,931	706,195	289,987
LATOUR1	1,663,550	680,237	270,423

### 3.3 Discussion of Results

In this discussion, the term "composite Huffman code" means any one of the following: T.81 - JPEG, D10, or Delta. In all three cases, a code word is assigned to every possible symbol. The term "optimized Huffman code" means a Huffman code optimized to a specific image. Only symbols that occur within that image are assigned code words. The term "other image Huffman code" means a Huffman code for a specific image, like the optimized code, except that a count of 1 is assigned to each *possible* symbol that never occurs in that image. This allows other images to be coded with the "other image code" for a given image. Since Huffman code is the only code employed in this study, the word "code" in the following discussion implies Huffman code.

For each test image and compression scale factor, the percentage difference by which the greatest exceeded the least bit count resulting from the use of the three composite codes was noted. The percentage difference by which the least bit count of the three composite codes exceeded the bit count resulting from the optimized code was also calculated. Finally, the percentage difference between the greatest and least bit counts resulting from employing the other Huffman image codes was computed.

For a compression scale factor of 9, D10 was the best of the three composite codes for seven of eight images, with T.81 - JPEG being the best in the other. However, the difference between the worst and the best never exceeded 3.6 percent. Therefore, for this scale factor, one can conclude that all three composite codes perform approximately equally well. The spread between the worst and best performances of the other image codes was considerably greater, ranging from 5.5 to 11.6 percent over the 8 images being compressed. The other image code derived from Image C1LAB gave the best performance in 7 of 8 cases, and performed approximately as well as, and sometimes better than, the composite codes. The best composite code bit count exceeded that of the optimized case by 0.9 to 2.4 percent over the eight test images.

For a compression scale factor of 24, the Delta code was best for five of the test images, and D10 was best in the other three. The difference between the worst and best ranged from 1.2 to 4.6 percent. The spread among the three composite codes is thus greater than with a compression scale factor of 9. The spread between the worst and best other image codes ranged from 3.1 to 7.8 percent. TOYSLAB and LATOUR2 each were best in two cases; no other image

was best in more than one case. The smallest bit count produced by the composite codes was 0.6 to 5.5 percent greater than that generated by the optimized code over the test image set.

For a compression scale factor of 71, the Delta code was the best of the three composite codes in all eight cases. The spread between the best and worst composite codes ranged from 0.95 to 11.4 percent depending upon the image being compressed. The spread between the worst and best other image codes was 1.9 to 28.6 percent. The performance of the best composite code with respect to the optimized code ranged from 1.2 to 15.5 percent.

Thus, the performance spreads between composite and optimized codes and among the different composite codes increased as the compression scale factor increased.

### 3.4 Conclusions

The foregoing results indicate that optimizing the Huffman code gives marginally better performance than a default code for low compression (high image quality), but considerably better performance when the data compression is high.

The notion of entropy, which is the lower bound on the average code word length for independently coded symbols, is based on the assumption of an inherent probability function for the symbol set. Section 4.0 shows the relationships among entropy, information, and Huffman codes. Entropy theory says, in effect, that the optimal *default* Huffman code is that which is derived from this probability function. In the long run, this Huffman code will perform as well as or better than any other *fixed* Huffman code. To approximate this inherent probability function, assuming that one exists, one should compile, for each of DC luminance, AC luminance, DC chrominance, and AC chrominance, a composite histogram of symbol occurrences over a *very large* number of images with widely varying image characteristics. The Huffman codes derived from these histograms should then be used as default codes. Of course, optimized codes will perform, in general, better than these default codes.

It is concluded that the composite codes employed in these experiments were compiled from an insufficient number of samples, or there was an insufficient mix of image characteristics that possess their own peculiar statistics. This would account for the considerable spread among the performances of the three composite codes, and an even greater spread among the "other image" codes.

It is also concluded that if default Huffman codes are employed, there should be a separate set for each combination of transmission parameters such as sub-sampling and data compression scale factor. The inherent probability functions

mentioned above probably vary sufficiently with transmission parameters to warrant separate code sets.

If image characteristics could be classified, as well as transmission parameters, then a separate Huffman code set could be generated for each class. The transmitter and receiver would both store all default Huffman code tables. The transmitter would decide which to use, based on transmission parameters and image class, and transmit the appropriate tables. If such classification produces default codes that are nearly optimal for all images in a given class, then optimized codes, which require the transmitter to gather statistics and generate the coding tables, would rarely be required.

## 4.0 HUFFMAN VS. INFORMATIONAL CODING

### 4.1 Introduction

There is a close parallel between theoretical coding, based on information values, and Huffman coding. While the former is not, in general, realizable in practice, it is very easy to treat analytically. Huffman coding produces average code word lengths that are typically within tenths of a bit, and are guaranteed to be within one bit, per coded symbol of the theoretical minimum.<sup>[1]</sup> Therefore, if one computes the theoretical minimum analytically, one obtains a good, although optimistic, estimate of the number of bits per symbol achievable with Huffman coding.

The following discussion shows the similarity between information and Huffman coding, including the notion of "optimal" and "default" information values that are the information counterparts of optimal and default Huffman codes.

### 4.2 Alphabets

Texts describing variable length coding typically refer to the transmission of "messages" composed of "symbols." The entire set of symbols from which a message can be composed is called an "alphabet." The information value, in bits, of each symbol,  $s$ , is given by:

$$I(s) = -\log_2 [p(s)],$$

where  $p(s)$  is the probability of symbol  $s$ . The entropy of a source producing such messages is given by:

$$H = \sum_s p(s) I(s) = -\sum_s p(s) \log_2 [p(s)]$$

This is the theoretical lower bound on the average number of coded bits per symbol when the symbols are coded independently of one another.

In the current context, there are four alphabets corresponding to the four sets of data for DC luminance, AC luminance, DC chrominance and AC chrominance. Since there are only two sets of symbols: SSSS for DC and RRRSSSS for AC, one might argue that there are only two alphabets. Because the statistics of luminance and chrominance data are distinctly different in both the DC and AC symbol sets, and because separate Huffman coding tables are provided for luminance and chrominance, the four sets of data are treated separately, and each is treated as having its own alphabet. It is also assumed that an alphabet is

comprised only of symbols that actually can occur, even if some do so only very rarely.\* The following discussion applies to any one of these alphabets.

### 4.3 Analysis

Consider the following two equations, the first applying to information coding and the second to Huffman coding.

$$L_i = \sum_s p(s) l(s) \quad (1)$$

$$L_h = \sum_s p(s) \quad (2)$$

where  $p(s)$  is a probability function,  $l(s)$  is defined by:

$$l(s) = -\log_2 [q(s)] \quad (3)$$

in which  $q(s)$  is *another* probability function that *may be, but is not necessarily, the same as*  $p(s)$ , and  $h(s)$  is a Huffman code word length.  $L_i$  and  $L_h$  are the average number of bits per symbol for information and Huffman coding respectively.

Equation (1) has the form of the entropy equation. However,  $l(s)$  is based on a probability function that is not necessarily  $p(s)$ . In the current context, the symbols of an alphabet do not have an inherent probability function as do the results of coin tosses, dice rolls and the dealing of poker hands. The relative occurrence frequencies of the symbols in an alphabet depend upon such parameters as sub-sampling and compression scale factor. Even with constant parameters, the relative frequencies vary from image to image.

For the purposes of this discussion, we assume that a probability function is estimated in the following manner: Accumulate a histogram of the number of occurrences,  $n(s)$ , of each symbol in an alphabet over one image, or over many images, and divide each  $n(s)$  by the total number of occurrences of all the symbols. The resulting set of ratios has the following required characteristics of a probability function: (1) each ratio lies in the closed interval of 0 through 1, and (2) the sum of the ratios is 1.

The only difference between such an estimated probability function and a theoretical one, aside from the fact that the estimated probabilities are approximate, is that a possible but improbable symbol might not occur at all in the accumulated data, in which case the estimated probability of the symbol is 0. In a

---

JPEG assumes 256 AC symbols, some of which are impossible. This study excludes impossible symbols from its "alphabets."

theoretical discrete probability function, the probabilities of all possible symbols are greater than 0.

The presence of zero probabilities in an estimated probability function means that in the accumulated data some rare but possible symbols simply failed to occur. The information value,  $I(s)$ , of a symbol having zero probability does not exist ( $-\log_2(p)$  approaches infinity as  $p$  approaches 0), and the JPEG simulation program does not generate a Huffman code for non-occurring symbols. This is not a problem if there is no symbol  $s$  for which  $p(s) > 0$  while  $q(s) = 0$ , because, in the theoretical case,  $-p \log_2(p)$  approaches 0 as  $p$  approaches 0, and in the Huffman case, a code word is not required for a symbol that never occurs. If, however,  $q(s)$  is the probability function from which "default" information values or Huffman codes are generated, then a symbol might occur in the histogram which generates  $p(s)$  but not in the one from which  $q(s)$  is derived. Consequently, when the latter is to be the basis of "default" codes, to each (possible)  $s$  for which  $n(s) = 0$  the program that builds the histogram of  $n(s)$  arbitrarily assigns a value of 1 to  $n(s)$ , thus preventing a zero value for  $q(s)$ .

It is now shown that, assuming that  $p(s) > 0$  and  $q(s) > 0$  for all  $s$ ,  $L_i$  in Equation (1) is minimum when  $q(s) = p(s)$ . The proof consists of minimizing, with respect to  $q_1, q_2, \dots, q_i, \dots, q_n$ , the function

$$F(q_1, q_2, \dots, q_i, \dots, q_n) = -[p_1 \ln(q_1) + p_2 \ln(q_2) + \dots + p_n \ln(q_n)]$$

with the constraint that the sum of the  $q$ 's is 1. The method is explained, for example, in Kaplan.<sup>[2]</sup> In  $F$ , it is valid to use natural logarithms instead of logarithms to the base 2, because  $\log_2(x) = \ln(x) / \ln(2)$ . Consequently, the  $q_i$ 's that minimize  $F$  also minimize  $L_i$ .

For the case at hand, let  $G(q_1, q_2, \dots)$  be the constraint function expressed as:

$$G(q_1, q_2, \dots, q_n) = q_1 + q_2 + \dots + q_n - 1 = 0.$$

Finding a critical point in the  $n$ -dimensional space consists of, for each  $i = 1, 2, \dots, n$ , adding the partial derivative of  $F$  with respect to  $q_i$  to a constant,  $m$ , times the partial derivative of  $G$  with respect to  $q_i$  and setting the result to zero. The resulting equation for each symbol  $i$  is:

$$-p_i / q_i + m = 0, \quad (4)$$

whence

$$q_i = p_i / m.$$

Putting in the constraint that the sum of the  $q$ 's is 1 gives:

$$\sum_i q_i = (1/m) \sum_i p_i = 1 \quad (5)$$

Since the sum of the  $p$ 's is also 1, Equations (4) and (5) are satisfied when and only when  $m = 1$  and  $q_i = p_i$  for all  $i$ . Thus, the critical point is the  $n$ -dimensional point  $q_i = p_i$  for all  $i$ . The fact that this point is a minimum is established by taking the second partial derivatives of  $F$  with respect to  $q_i$ . When  $q_i = p_i$ , the second derivatives are  $1/p_i$ , which, for  $p_i > 0$ , exist and are positive.

#### 4.4 The Comparison

For each alphabet, let  $p(s)$  now be defined as the estimated probability function derived from the histogram compiled from the transmission of *one image*, which will be called the test image. Let  $q(s)$  be the estimated probability function derived from a histogram which is compiled from one of the following: (1) the test image, (2) some other image, or (3) a composite of a number of images which may but need not include the test image. Except for the special case in which the histogram for  $q(s)$  is compiled from just the test image, this histogram must be adjusted, if necessary, to guarantee that  $n(s) > 0$  for all (possible)  $s$ , as described above, to ensure that there is no  $s$  for which  $p(s) > 0$  while  $q(s) = 0$ . Let  $h(s)$  be the Huffman code word length for symbol  $s$  derived from the *same histogram* that produces  $q(s)$ . Then  $L_i$ , as defined in Equation (1), is a good predictor of the average Huffman code word length per symbol,  $L_h$ , as defined in Equation (2), and, as experiments described below show, is a *better*, more realistic approximation than the optimistic value given by the entropy.  $l(s)$  in Equation (1) can be thought of as the information code word length, analogous to  $h(s)$ , the Huffman code word length in Equation (2). When  $q(s) = p(s)$ , the  $l(s)$  values are optimal, and  $L_i$  is minimum, just as an optimized Huffman code derived from  $p(s)$  gives the minimum average Huffman code word length. If  $q(s)$  is derived from a composite histogram of many images, as is typically done to generate a "default" Huffman code, then both  $L_i$  and  $L_h$  increase. Thus, one can think of  $l(s)$  as "optimal" or "default" information values for  $q(s)$  equal or not equal to  $p(s)$ , analogously to "optimal" or "default" Huffman code word lengths.

#### 4.5 Experiments

The purpose of the experiments was to observe the behavior of the average information value and Huffman code word length when a test image is coded with optimized and non-optimized information values and Huffman code words.

The experiments were performed on the following data, one data set per alphabet:

- (1) A histogram of symbols that actually occur for each of eight test images,

- (2) A histogram for each image with possible, but non-occurring symbols assigned counts of 1,
- (3) A composite histogram consisting of the sum of the histograms for all eight images, with possible symbols that never occur in any of the eight images assigned counts of 1.

The test images are fully sampled and compressed with a compression scale factors of 25, which is the JPEG default setting.

Probability functions  $p(s)$  were computed for each image from histograms (1). Probability functions  $q(s)$  were computed from histograms (2) and (3). Histograms (2) and (3) were submitted to the modified JPEG simulator to obtain Huffman code length data from each histogram for all the possible symbols.

For each alphabet,  $L_i$ , the average information value, was computed by Equation (1) for each test image and for each of the  $q(s)$  functions derived from histograms (2) and (3) to show how  $L_i$  behaves when the source of coding information (called the "code source" in the following tables) is the same image, a different image, or the composite. Similarly, the average Huffman code word length,  $L_h$ , was computed by Equation (2) for each test image and for each Huffman code word length table derived from histograms (2) and (3).

It should be noted that, for any given image, the total number of coded bits for all four alphabets is less than the total number of bits in the compressed bit stream. The latter is comprised also of SSSS bits per symbol, plus overhead, where SSSS is the "size" component of the symbol. The sensitivity of  $L_i$  and  $L_h$  to different code sources is therefore masked somewhat by the presence of these other bits; hence the sensitivity of the overall data compression is less than that of  $L_i$  or  $L_h$ .

#### 4.6 Results

Table 1 gives the names of the test images to which the test image and code source numbers in the remaining tables correspond. Code source 9 is derived from the composite histogram of all eight test images, and is therefore not considered as a single test image. Images 4 and 5, LATOUR1 and LATOUR2, are actually the left and right halves of the LATOUR image, which was too large to process through the JPEG simulator.

Tables 2 through 5 show the experimental results for the four alphabets. Each row represents different test images coded by the same code source, and each column represents a single test image coded by different code sources. In each table cell, the top value is the average information value, and the bottom value is the average Huffman code length.



**TABLE 1**  
**KEY TO TEST IMAGES AND CODE SOURCES**

Number	Test Image / Code Source
1	APPOTLAB
2	C1LAB
3	FAXBALLS
4	LATOUR1
5	LATOUR2
6	PM003_NL
7	SA001_NL
8	TOYSLAB
9	Composite (code source only)

#### 4.7 Observations and Discussion

In each table column (same test image, different coding sources), both the average information value and Huffman code word length are minimum, as expected, when the coding source is the same image, yielding optimized information values and Huffman codes, i.e.,  $q(s) = p(s)$ .

When the code source is other than the test image, the average information value computed from Equation (1) is, in most but not all cases, a better predictor of the average Huffman code word length than the entropy, which is the average information value when the code source is the test image. In a large number of cases,  $L_i$  and  $L_h$  agreed to within hundredths of a bit per symbol. In some cases the values differed by a few tenths of a bit per symbol.

The composite histogram, in most, but not all cases, produced Huffman codes that were at least as efficient as Huffman codes produced by any single code source other than the test image itself.

The Huffman code length averages in rows 4 and 5 (LATOUR1 and LATOUR2) of the DC luminance table are identical for a given test image, although the average information values are different, different by approximately two tenths of a bit per symbol in column 7. Further investigation revealed that these two images produced identical Huffman code word lengths for each DC luminance symbol despite considerably different histograms. Thus, Huffman code word lengths are less sensitive to differences in probability functions than are information values.

In a number of cases, but never when the coding source was the test image, the average Huffman code word length was less than the average information value. This result was so counter-intuitive (because the average information formula closely resembles the entropy formula) that one of these cases in the DC luminance alphabet was verified by hand from the raw data (with the aid of a spreadsheet program) to prove that the result was real, and not due to a bug in the computer programs. Evidently, although the entropy ( $q(s) = p(s)$ ) is the absolute lower bound on both the average information value and average Huffman code word length, the average information value computed when  $q(s)$  is different from  $p(s)$  is not a lower bound on the average Huffman code word length when the Huffman code is derived from  $q(s)$ .

#### 4.8 Conclusion

Averages of information values derived from the same probability function as that which produces a Huffman code are easy to compute without actually generating Huffman codes, and are good predictors of Huffman code performance. When this probability function is the same as that of a test image, the average information is the entropy, and the resulting Huffman code is optimal for that image. When the probability function is different from that of the test image, the resulting average information value and Huffman code word length both increase, the former sometimes increasing more than the latter, but the two are generally in better agreement than are the degraded Huffman code and the entropy.

**TABLE 2**  
**AVERAGE INFORMATION VALUES AND**  
**HUFFMAN CODE LENGTHS FOR DC LUMINANCE**

Test Image

Code Source		1	2	3	4	5	6	7	8
	1	2.4851 2.5857	3.5812 3.3677	3.3456 3.0596	2.5169 2.5499	2.6887 2.6584	2.6606 2.6656	3.4370 3.1387	3.0655 2.9281
	2	3.0786 3.1395	2.9656 3.0526	2.9775 3.0726	3.0750 3.1601	3.0556 3.1503	3.0102 3.0317	2.9685 3.1119	3.0117 3.1197
	3	3.2784 3.6471	3.3948 3.6938	2.6314 2.6710	3.0059 3.2593	2.9931 3.2209	2.9134 3.1591	2.8779 2.9741	3.0490 3.2579
	4	2.6047 2.6888	3.8285 3.8553	3.1357 3.1189	2.4200 2.4554	2.5533 2.5696	2.7001 2.8008	3.1704 3.1056	2.9682 2.9641
	5	2.6579 2.6888	3.6916 3.8553	2.9919 3.1189	2.4424 2.4554	2.5300 2.5696	2.7206 2.8008	2.9704 3.1056	2.8728 2.9641
	6	2.6194 2.8733	3.4208 3.6378	2.9964 3.0261	2.5947 2.8111	2.7266 2.9542	2.5267 2.6232	3.2636 3.4861	3.0135 3.2457
	7	3.2903 3.3140	3.4415 3.3779	2.9489 3.0372	3.0021 3.0576	2.9119 2.9704	3.2852 3.3750	2.6299 2.6968	2.9233 2.9637
	8	2.7459 2.7887	3.2478 3.4369	2.8500 3.0255	2.5973 2.6167	2.6182 2.6416	2.7791 2.9410	2.7612 2.8359	2.7738 2.8407
	9	2.6190 2.6273	3.2625 3.4181	2.8879 2.8900	2.5218 2.5115	2.5867 2.5912	2.6530 2.6347	2.8950 2.9731	2.7995 2.8566

**TABLE 3**  
**AVERAGE INFORMATION VALUES AND**  
**HUFFMAN CODE LENGTHS FOR AC LUMINANCE**

**Test Image**

**Code Source**

	1	2	3	4	5	6	7	8
1	3.3138 3.3501	3.6623 3.6663	3.3955 3.4344	3.1771 3.2292	3.1872 3.2348	3.8153 3.8417	3.3160 3.3514	3.2420 3.2814
2	3.5251 3.5602	3.4469 3.4839	3.4555 3.4060	3.4518 3.4526	3.4120 3.4145	3.7836 3.8346	3.2717 3.2709	3.4038 3.4092
3	3.5668 3.5520	3.6724 3.7092	3.1553 3.1986	3.3465 3.3305	3.3649 3.3579	3.7772 3.7974	3.3627 3.3978	3.3923 3.3966
4	3.3492 3.4202	3.7641 3.9051	3.3608 3.3818	3.1395 3.1781	3.1651 3.2166	3.8963 3.9965	3.3398 3.4312	3.2437 3.3015
5	3.3425 3.4196	3.7160 3.9106	3.3540 3.3760	3.1449 3.1839	3.1599 3.2183	3.8817 4.0132	3.2996 3.4305	3.2306 3.2987
6	3.4598 3.4821	3.5465 3.5403	3.2622 3.3211	3.3140 3.3573	3.3217 3.3551	3.6492 3.6751	3.3186 3.3266	3.3395 3.3662
7	3.4407 3.5643	3.5511 3.5454	3.3576 3.4297	3.2901 3.4354	3.2644 3.3926	3.8644 3.9115	3.1993 3.2452	3.2765 3.3893
8	3.3429 3.3859	3.6421 3.6388	3.3295 3.3742	3.17563 3.2349	3.1786 3.2294	3.8362 3.8867	3.2612 3.2794	3.2168 3.2601
9	3.3519 3.4012	3.5306 3.5566	3.3024 3.3200	3.2091 3.2586	3.20643 3.2543	3.7280 3.7713	3.2317 3.2609	3.2394 3.2809

**TABLE 4**  
**AVERAGE INFORMATION VALUES AND**  
**HUFFMAN CODE LENGTHS FOR DC CHROMINANCE**

Test Image

Code Source

	1	2	3	4	5	6	7	8
1	1.9022 1.9845	3.0695 2.9388	2.3880 2.2906	1.8629 1.8728	2.1101 2.1322	1.9979 1.8893	2.1539 2.2379	2.1060 2.1775
2	2.2928 2.1654	2.5533 2.6391	2.3350 2.3011	2.2262 2.1250	2.2864 2.1920	2.2242 2.1897	2.3138 2.1544	2.2984 2.1636
3	2.0955 2.2388	2.7154 2.9496	2.1908 2.2689	1.9755 2.0782	2.1394 2.2990	1.9511 1.9914	2.1855 2.3486	2.1655 2.3398
4	1.9649 1.9845	3.1546 2.9388	2.3779 2.2906	1.8258 1.8728	2.0881 2.1322	1.9217 1.8893	2.1785 2.2379	2.1088 2.1775
5	1.9669 1.9845	2.8954 2.9388	2.2889 2.2906	1.8678 1.8728	2.0438 2.1322	1.9692 1.8893	2.0779 2.2379	2.0391 2.1775
6	2.0434 1.9845	2.9630 2.9369	2.2917 2.2906	1.8833 1.8728	2.1504 2.1322	1.8619 1.8891	2.2915 2.2379	2.2175 2.1775
7	2.0179 2.1654	2.9592 2.6410	2.3477 2.3011	1.9682 2.1250	2.0934 2.1920	2.1260 2.1898	2.0272 2.1544	2.0353 2.1636
8	1.9909 1.9974	2.9647 2.9942	2.3374 2.5100	1.9112 2.0080	2.0600 2.1614	2.0585 2.2160	2.0410 2.1550	2.0230 2.1199
9	1.9481 1.9845	2.8150 2.9369	2.2610 2.2906	1.8752 1.8728	2.0499 2.1322	1.9644 1.8891	2.0800 2.2379	2.0469 2.1775

**TABLE 5**  
**AVERAGE INFORMATION VALUES AND**  
**HUFFMAN CODE LENGTHS FOR AC CHROMINANCE**

Test Image

Code Source

	1	2	3	4	5	6	7	8
1	2.5979 2.6858	3.5480 3.4500	2.5075 2.6954	2.4565 2.6227	2.7258 2.8066	3.4287 3.4582	2.5614 2.6514	2.5253 2.6235
2	2.8616 2.8034	3.1297 3.1584	2.5756 2.5139	2.6839 2.6081	2.8279 2.7860	3.1802 3.1494	2.7116 2.6789	2.7823 2.7257
3	2.9204 2.9014	3.4706 3.4833	2.2519 2.2717	2.4698 2.4700	2.8481 2.8499	3.2273 3.2091	2.7638 2.7763	2.8158 2.8126
4	2.6897 2.7343	3.5002 3.6255	2.3452 2.3830	2.3672 2.3928	2.7063 2.7555	3.2425 3.3855	2.6120 2.6438	2.5969 2.6369
5	2.6431 2.7280	3.3310 3.5765	2.3867 2.3637	2.4072 2.3895	2.6664 2.7441	3.1845 3.3143	2.55322 2.6511	2.5564 2.6382
6	2.8069 2.8578	3.2507 3.1879	2.3798 2.5717	2.4884 2.6498	2.7601 2.8240	3.0451 3.1180	2.7083 2.7414	2.73512 2.7891
7	2.6375 2.7231	3.4402 3.4087	2.4697 2.6515	2.4627 2.6113	2.7080 2.7974	3.3821 3.4519	2.5204 2.6113	2.5380 2.6326
8	2.6129 2.6960	3.6007 3.5679	2.5221 2.7026	2.4539 2.6220	2.7266 2.8205	3.5072 3.5409	2.5442 2.6330	2.5165 2.6034
9	2.6394 2.7293	3.2740 3.2090	2.3933 2.5501	2.4327 2.5796	2.6735 2.7486	3.1534 3.1941	2.5520 2.6248	2.5600 2.6526

## 5.0 FUTURE PLANS

### 5.1 Determining Optimal Default Huffman Code

The main conclusion reached from the Huffman coding study was that the tested "default" Huffman codes came from an insufficiently large sample of the Huffman coded symbols or that there was an insufficient mix of image characteristics.

#### 5.1.1 Criterion for Determining Required Sample Sizes

Suppose that a symbol  $s$  has probability  $p(s)$ , and one attempts to estimate the value of  $p(s)$  by sampling a large number of symbols at random. The smaller the value of  $p(s)$ , the larger must be the number of samples to obtain a reliable estimate of  $p(s)$ . If one takes  $N$  samples, where  $N$  is a large number, then the expected, but by no means necessarily the actual, number of times that symbol  $s$  occurs is  $N p(s)$ . For  $p(s)$  small, the standard deviation about  $N p(s)$  is given by

$$\sigma = \sqrt{N p(s)} .$$

For the number of occurrences of  $s$  to have a good chance of being within some fraction,  $f$ , of  $N p(s)$  requires that the standard deviation be  $f N p(s)$ , from which

$$N > 1 / [p(s) f^2].$$

Thus, the smaller  $p(s)$ , the larger the number of samples required. For example, if  $p(s) = 0.001$ , then 100,000 symbols would have to be sampled for there to be a good chance (about 68% probability, i.e. within one standard deviation of a normal probability density function) for symbol  $s$  to occur within 10 percent of the expected number of times.

The probability function  $p(s)$  itself, however, depends upon the mix of the image characteristics in the collection of images from which the samples are drawn. Different kinds of images have different probability functions; for example, busy vs. bland, line drawings vs. landscapes. One must in effect pour the symbols from a large number of images representing all kinds of images likely to be compressed into a common pool, and then create a composite histogram (for each alphabet) either from the entire pool or from sufficiently many random samples taken from that pool, where the number of samples required is dictated by the above sampling criterion.

### **5.1.2 Approach to Best Default Huffman Codes**

The first step is to collect as many test images as possible to produce a thorough mix of the various image characteristics. Ideally, the mix of image types in the collection would be in the same proportions as in the "universe" of all images ever transmitted. If this "universal mix" cannot be estimated in any straightforward manner, then one must simply collect, at random, as many different kinds of images as possible.

The next step is to pour randomly chosen subsets of the available images into separate pools, and build composite histograms from each pool as well as from the entire set. If the Huffman codes generated from the subsets perform to within a few percent of one another and of the entire set, then one can conclude that the various image types are sufficiently well represented in the collection.

If a symbol pool is too large to generate a composite histogram or a Huffman code, one can employ the theory presented in Section 4.0 to determine the sensitivity of the average information to errors in probability estimates. This sensitivity function and the sampling criterion given above determine the required sample size.



## REFERENCES

- [1]. Aron N. Netravali and Barry G Haskell, *Digital Pictures Representation and Compression*, page 150, Plenum Press, New York and London
- [2]. Wilfred Kaplan, *Advanced Calculus*, p. 129, Addison-Wesley Publishing Company, Inc., Cambridge, MA

## Appendix A

### Image Histograms

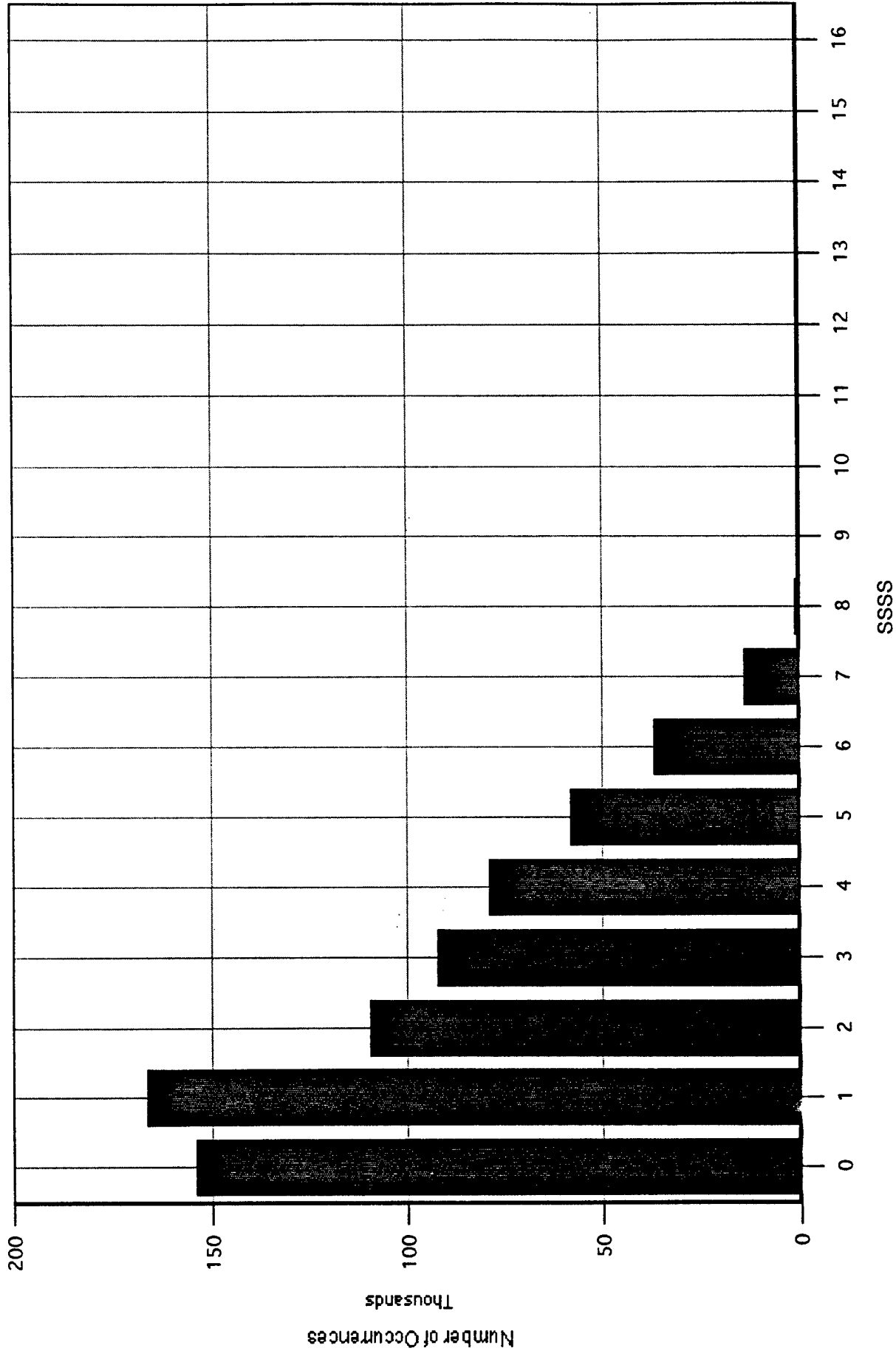
## IMAGE HISTOGRAMS

The following plots are of histograms of the four Huffman coded symbol sets: DC luminance, AC luminance, DC chrominance, and AC chrominance. The composite histograms and those of the FAXBALLS image are shown. The histograms apply to fully sampled images and a compression scale factor of 24. The composite histogram was used to generate the Delta Huffman code, which was one of the "default" Huffman codes tested in experiments reported in Section 3.0.

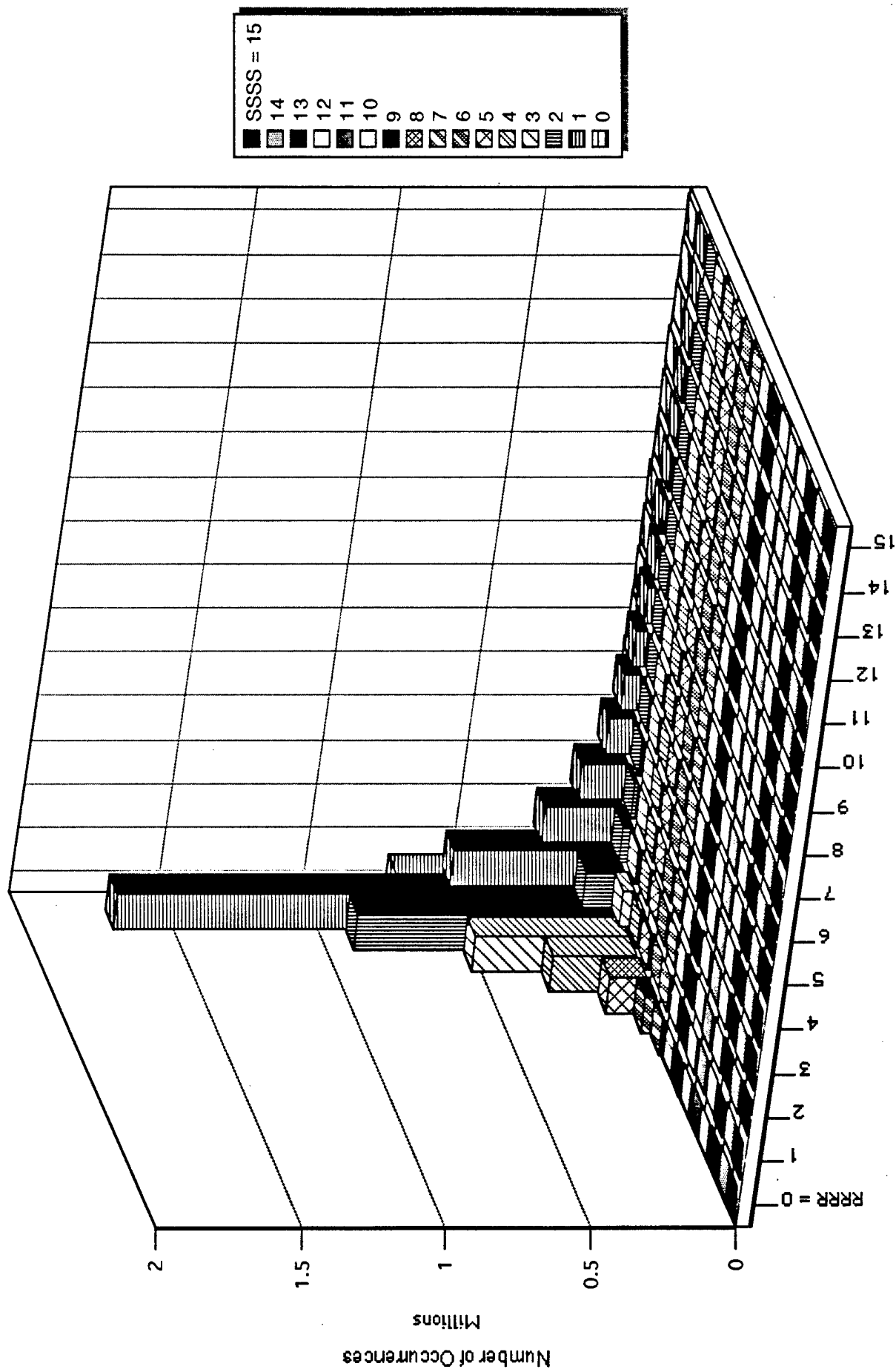
The FAXBALLS image was chosen because the percentage difference, 4.6 percent, between the compressed bit count produced by the Delta Huffman code and by the optimized Huffman code was greater for FAXBALLS than for any other image in the test set. This selection shows how widely image histograms can differ from the composite and yet produce only a few percent degradation in data compression. Actually, the percentage difference in bit count owing to the Huffman coding alone was 6.3 percent; the 4.6 percent figure is based on the *total* bit count, to which the Huffman coding contributed approximately 65 percent.

# Composite DC Histogram for Luminance

Total Occurrences: 709198

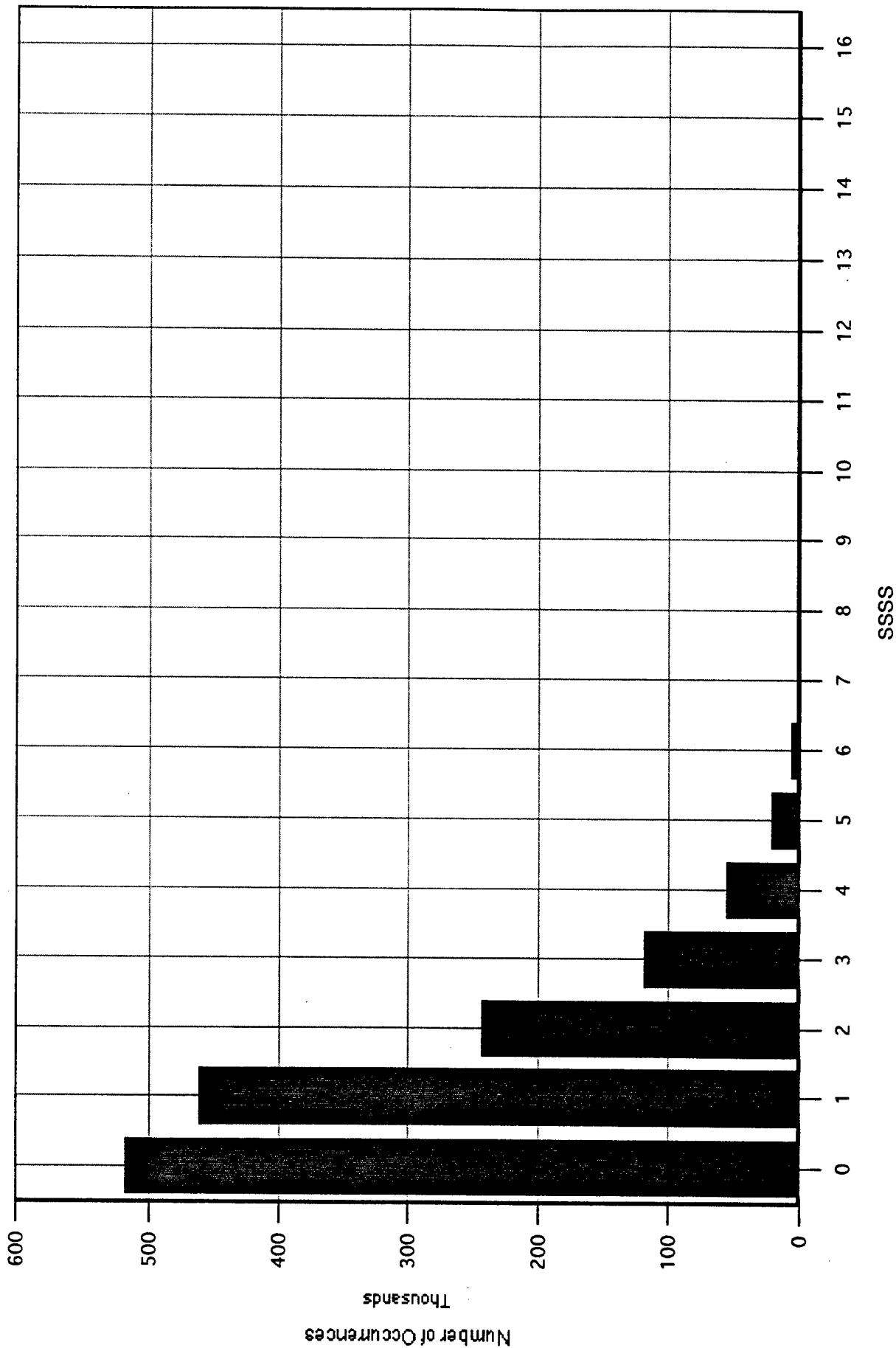


**Total Occurrences: 5859738**

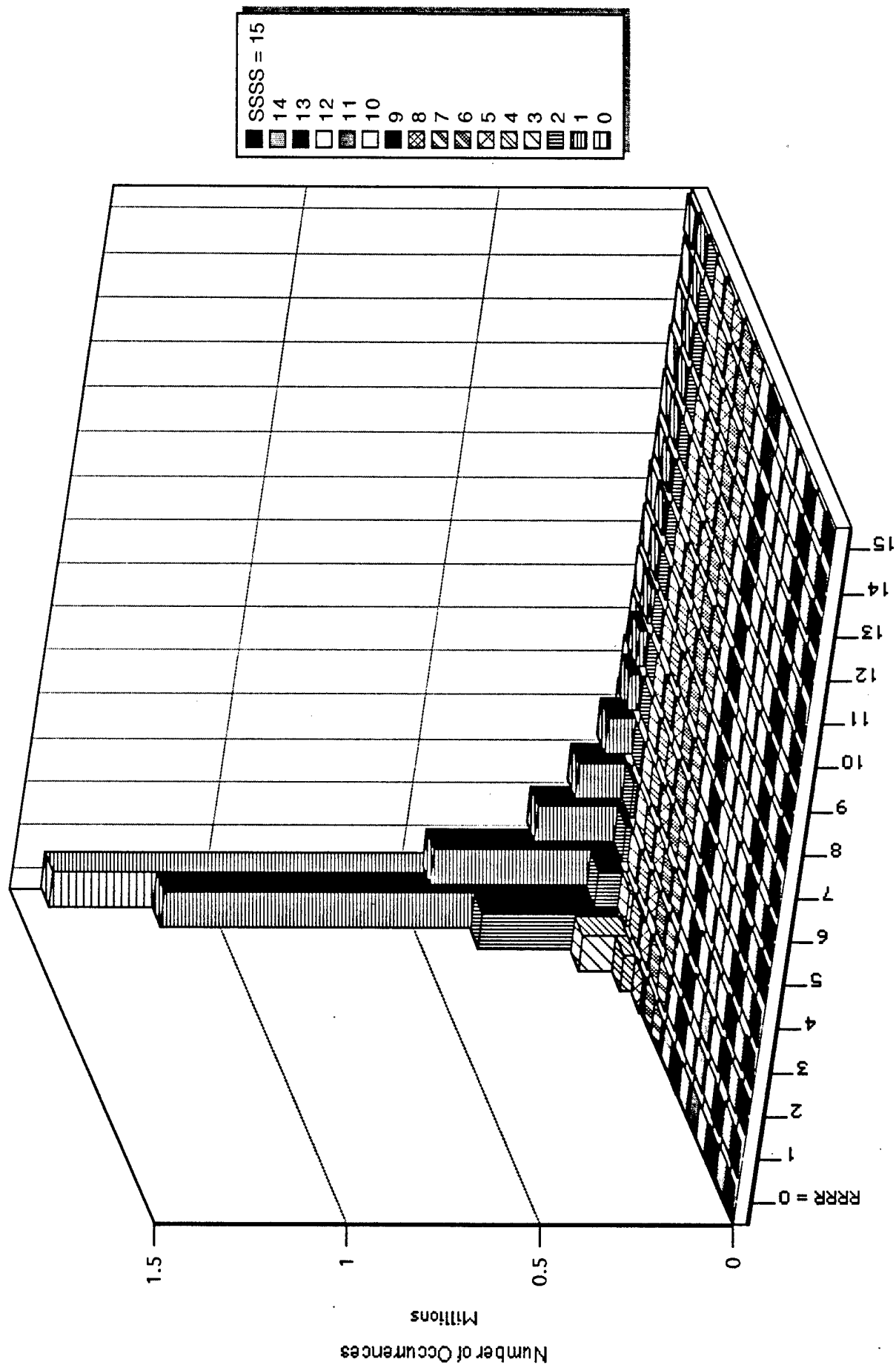


# Composite DC Histogram for Chrominance

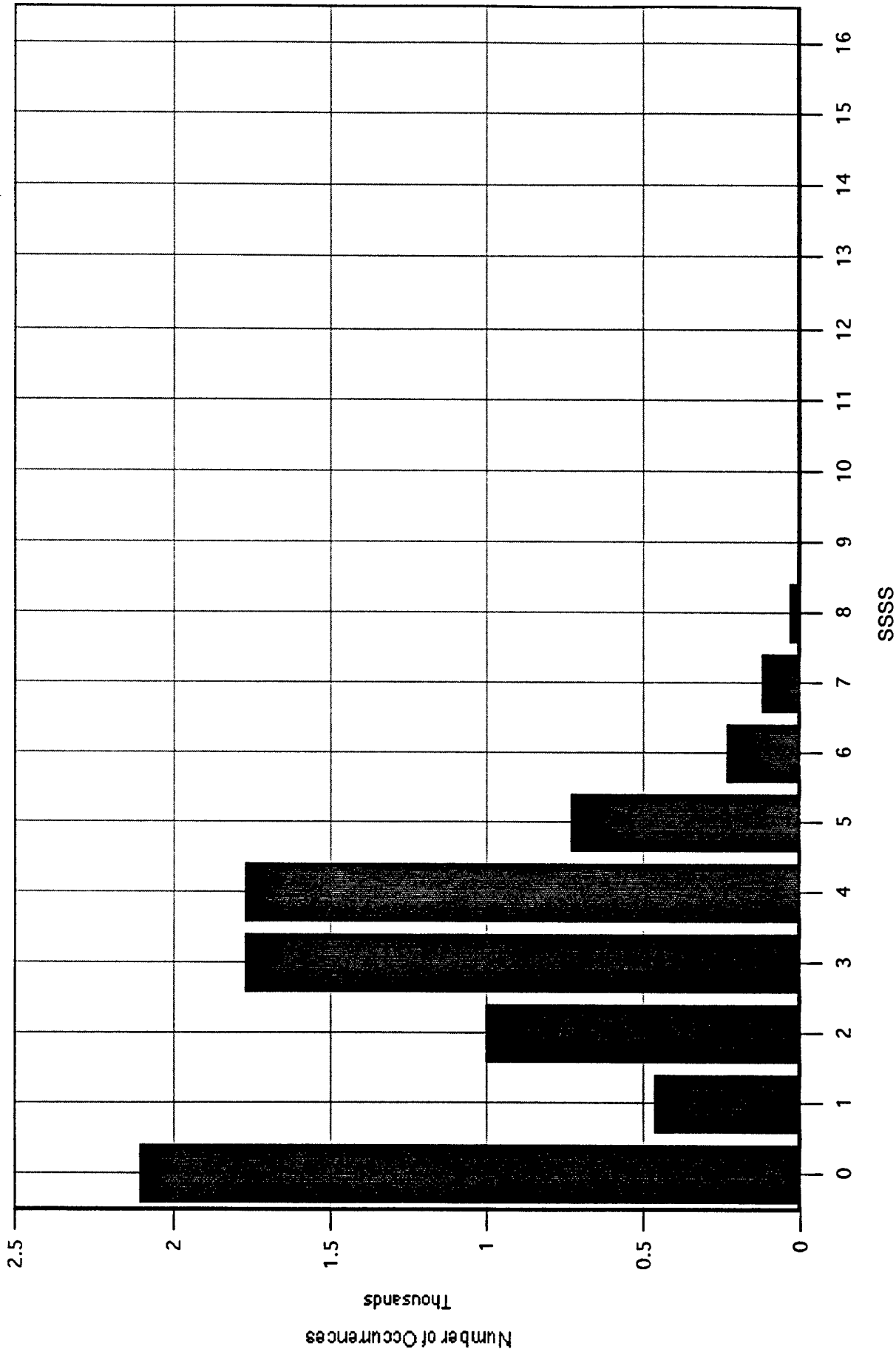
Total Occurrences: 1418393



**Total Occurrences: 4158865**

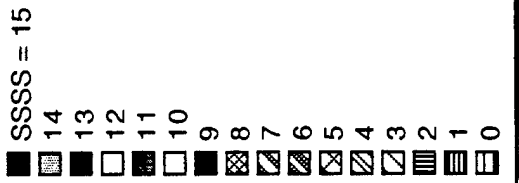


DC Histogram of FAXBALLS Luminance  
Total Occurrences: 8192



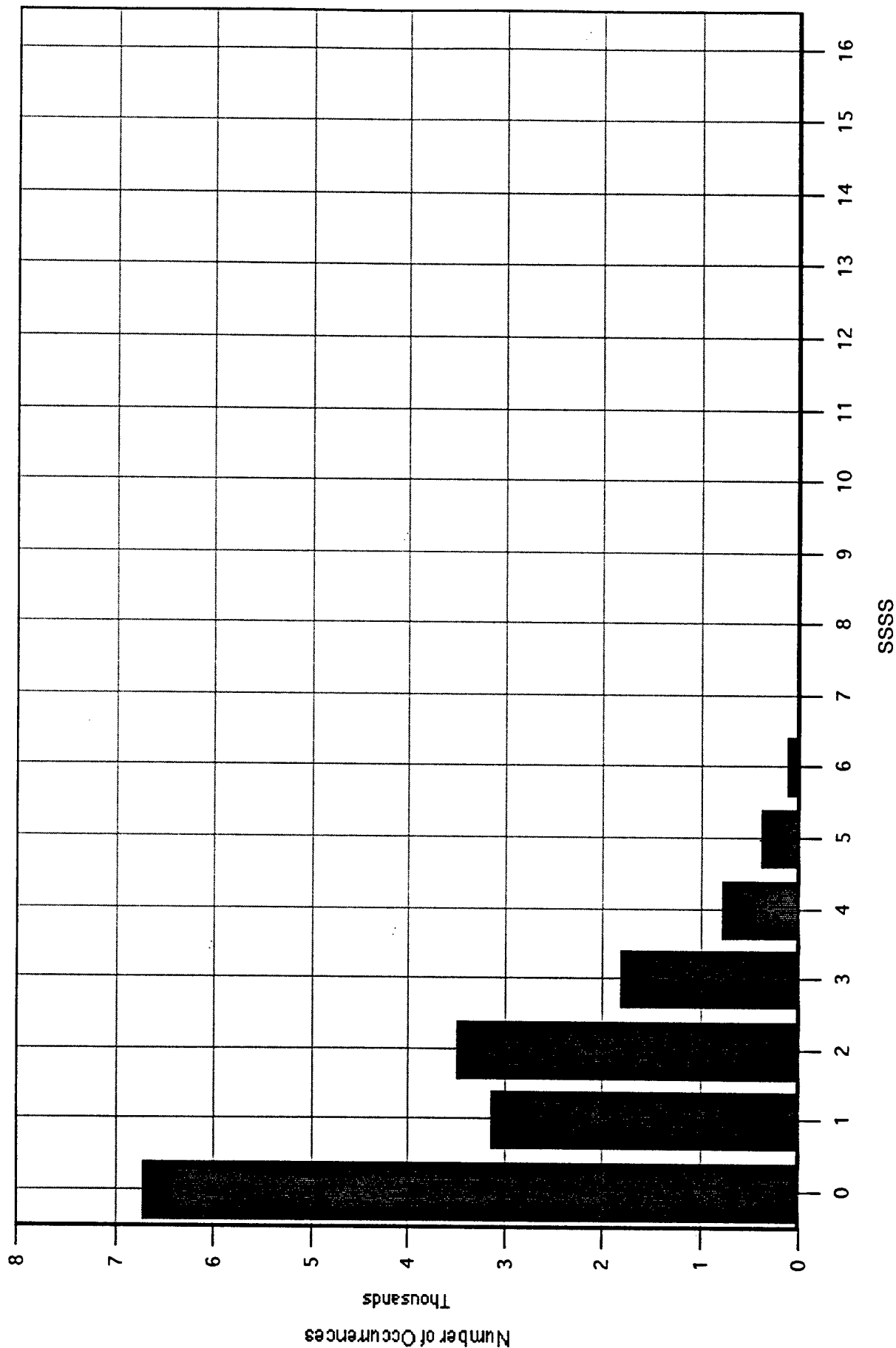


**Total Occurrences: 35290**



# DC Histogram of FAXBALLS Chrominance

Total Occurrences: 16384



## Total Occurrences: 33198



## Appendix B

### Contribution D10 - Study Group 8

Commission d'études  
Study Group  
Comisión de Estudio

} 8

Contribution tardive  
Delayed Contribution  
Contribución tardía

} D 10

Genève, 27 avril - 6 mai 1993

Geneva, 27 April - 6 May 1993

Ginebra, 27 de abril - 6 de mayo de 1993

Texte disponible seulement en

Text available only in

Texto disponible solamente en

} E

Questions: 4, 16/8

SOURCE: JAPAN

Title: Experimental result for deciding JPEG default Huffman tables  
on the color facsimile standardization

## 1. Introduction

In the associated rapporteur group meeting for Color Extension for Group4 Facsimile held in November 1992, it is decided that in the case where the Huffman tables are not transmitted by the sending terminal, default Huffman tables must be used. Under this decision, Japan has been carried out an experiment for providing default Huffman tables. This contribution shows a result of our experiment.

## 2. Discussion and Proposals

### 2.1 Experimental procedure for deciding default Huffman tables

#### 1) Test Images

- High Resolution Digital Test Image (NTT)  
pm003\_nlab.0~2(2048\*2048)  
sa001\_nlab.0~2(2048\*2048)
- SCID Test Image (ISO TC130)  
Cafeteria(2048\*2560)

#### 2) Color Space

CIE LAB

#### 3) Subsampling

(1:1:1) and (4:2:2)

In the case of (4:2:2), 1/2 subsampling is carried out to "a" and "b" color components after horizontal low-pass-filtering weighted 1:2:1. Filtering result is calculated by rounding.

Sample values which are suited outside of the image boundary are replicated from the sample values at the boundary to provide missing edge values in filtering calculation.

After filtering, subsampling is realized by taking odd numbered pixel from left edge in one pixel line.

#### 4) Quantization

Quantization step size =

{ Recommended Quantization Table \* Scaling factor / 50 }  
(Round to integer)

Recommended Quantization Tables are shown at page 161 in ISO DIS 10918-1.

(L → Table K.1. a·b → Table K.2)

Scaling factors used for above calculation are as follows.

9 → 2.0bit/color-pixel

24 → 1.0bit/color-pixel

71 → 0.5bit/color-pixel

#### 5) Weighting Function

Weighting Function  $W_j$  ( $j=1\sim6$ ,  $\sum W_j=1.0$ ) is defined for combination of 2 subsampling rates and 3 scaling factors as follows.

Scaling factor

Subsampling rate	Scaling factor		
	9	24	71
	J=1	J=2	J=3
1:1:1	J=1	J=2	J=3
4:2:2	J=4	J=5	J=6

$W_1=2/12$ ,  $W_2=4/12$ ,  $W_3=2/12$

$W_4=1/12$ ,  $W_5=2/12$ ,  $W_6=1/12$

#### 6) Deriving Huffman Table

The procedure for deriving DC and AC default Huffman tables for L and a,b color components are shown in ANNEX A. The derived default Huffman tables are shown in ANNEX B.

## 2.2 Comparison of compression rate

Table.1 shows comparison of compression rates for following 3 Huffman tables.

- Optimized Huffman tables for each image (Custom tables)
- Default Huffman tables (ANNEX B)
- Recommended Huffman tables from JPEG

As a result, difference of Compression rate between default tables and recommended tables is very small.

When compression rate (bit/pel) is more than 1.0, difference is  $-0.8 \sim +1.8\%$ . ("-" means that recommended tables realize higher compression than default tables.)

When compression rate (bit/pel) is smaller than 1.0, difference is  $+1.2 \sim +3.8\%$ .

Table.1 Comparison of compression rate

Image: pm003

Top row : Optimized Huffman tables  
 Middle row : Default Huffman tables derived  
 Bottom row : Recommended Huffman tables  
 in T.81

	9	24	71
	Coded Data Bit Rate Amount(bits)(bits/pel)	Coded Data Bit Rate Amount(bits)(bits/pel)	Coded Data Bit Rate Amount(bits)(bits/pel)
1:1:1	8,490,219 2.024226	4,377,975 1.043791	2,246,747 0.535666
	8,596,292 2.049516	4,439,235 1.058396	2,475,670 0.590246
	8,656,885 2.063962	4,528,658 1.079716	2,556,416 0.609497
4:2:2	6,510,892 1.552318	3,484,759 0.830831	1,820,211 0.433972
	6,584,480 1.569862	3,518,823 0.838953	1,967,006 0.468971
	6,647,437 1.584873	3,593,834 0.856837	2,036,570 0.485556

Image: sa001

	9	24	71
	Coded Data Bit Rate Amount(bits)(bits/pel)	Coded Data Bit Rate Amount(bits)(bits/pel)	Coded Data Bit Rate Amount(bits)(bits/pel)
1:1:1	10,831,817 2.582506	5,679,529 1.354105	2,502,094 0.596546
	10,977,181 2.617164	5,765,274 1.374548	2,781,095 0.663065
	11,016,144 2.626453	5,795,731 1.381810	2,819,801 0.672293
4:2:2	8,748,416 2.085785	4,760,869 1.135080	2,192,600 0.522757
	8,863,644 2.113257	4,832,815 1.152233	2,377,852 0.566924
	8,848,053 2.109540	4,835,960 1.152983	2,405,835 0.573596

Image: Cafeteria

	9	24	71
	Coded Data Bit Rate Amount(bits)(bits/pel)	Coded Data Bit Rate Amount(bits)(bits/pel)	Coded Data Bit Rate Amount(bits)(bits/pel)
1:1:1	22,985,467 4.384130	12,946,460 2.469341	6,589,172 1.256785
	23,691,375 4.518771	13,035,925 2.486405	6,697,457 1.277439
	23,519,154 4.485923	13,039,429 2.487074	6,738,258 1.285221
4:2:2	18,831,903 3.591900	10,971,185 2.092588	5,719,126 1.090837
	19,410,365 3.702234	11,063,594 2.110213	5,790,657 1.104480
	19,253,435 3.672301	11,034,374 2.104640	5,809,481 1.108071



### 3. Conclusion

This contribution presents the experimental results for deciding default Huffman tables.

Default Huffman tables are derived from a statistical data from 3 test images, and compression rates by using this default tables are compared with the optimised Huffman tables for each image (custom tables) and recommended Huffman tables from JPEG.

As a result, difference of Compression rate between default tables and recommended tables is very small.

## ANNEX A

## Procedure for deriving default Huffman tables

- ① For each image, deriving  $PDJ(n)$  and  $PAJ(r,s)$  for each combination(j) of subsampling rate and scaling factor.

$PDJ(n)$ : Probability of DC difference magnitude categories  
 (see page 102 in DIS. 10918-1)  
 $n=0 \sim 11, \sum_n PDJ(n)=1.0$

$PAJ(r,s)$ : Two dimensional probability of AC run/size combination  
 (see page 104 in DIS 10918-1)  
 $r=0 \sim 15$  (Run:runlength of zero coefficients)  
 $s=0 \sim 10$  (Size:category of non-zero coefficient)  
 $(0,0) \rightarrow EOB$   
 $(15,0) \rightarrow ZRL$   
 $(1,0) \text{ } l=1 \sim 14 \text{ undefined}$   
 $\sum_{r,s} PAJ(r,s)=1.0$

- ② For each image, deriving weighted probability  $PD(n)$  and  $PA(r,s)$ .

$$PD(n) = \sum_j WJ \cdot PDJ(n)$$

$$PA(r,s) = \sum_j WJ \cdot PAJ(r,s)$$

- ③ Deriving averaged probability  $PPD(n)$  and  $PPA(r,s)$  among plural images.

$$PPD(n) = \sum_{\text{images}} PD(n) / (\text{number of images})$$

$$PPA(r,s) = \sum_{\text{images}} PA(r,s) / (\text{number of images})$$

- ④ Deriving BITS, HUFFVAL

Deriving BITS and HUFFVAL from  $PPD(n)$  and  $PPA(r,s)$ , according to Chapter K.2, page 181-188, in DIS 10918-1.

BITS: list of code length  
 HUFFVAL: list of values

© Deriving EHUFCO, EHUFSl.

Deriving EHUFCO and EHUFSl from BITS and HUFFVAL, according to Chapter C , page 83-88 , in DIS 10818-1.

EHUFCO: code table  
EHUFSl: code size table

# ANNEX B Derived default Huffman tables

```

*** L*          DC Huffman table          ***

** BITS in hex. **
00 01 05 01 01 01 01 01 01 00 00 00 00 00 00
** HUFFVAL in hex. **
00 01 02 03 04 05 06 07 08 09 0A 0B
Category      Code length  Code word
0              2           00
1              3           010
2              3           011
3              3           100
4              3           101
5              3           110
6              4           1110
7              5           11110
8              6           111110
9              7           1111110
10             8           11111110
11             9           111111110

```

```

*** a*, b*      DC Huffman table          ***

** BITS in hex. **
00 03 01 01 01 01 01 01 01 01 01 00 00 00 00
** HUFFVAL in hex. **
00 01 02 03 04 05 06 07 08 09 0A 0B
Category      Code length  Code word
0              2           00
1              2           01
2              2           10
3              3           110
4              4           1110
5              5           11110
6              6           111110
7              7           1111110
8              8           11111110
9              9           111111110
10             10          1111111110
11             11          11111111110

```

```

*** L*          AC Huffman table          ***

** BITS in hex. **
00 01 03 02 04 03 04 05 07 08 06 04 01 01 02 6F
** HUFFVAL in hex. **
01 00 02 03 04 11 05 12 21 31 06 41 51 13 22 61
71 07 14 32 81 91 15 23 42 52 A1 B1 C1 08 24 33
62 72 D1 E1 F0 16 34 43 53 82 92 25 35 63 73 A2
B2 F1 26 44 54 83 93 C2 17 36 B3 09 45 64 74 94
A3 D2 84 18 27 55 E2 F2 37 46 65 A4 B4 C3 28 85
C4 F3 56 75 86 B5 D3 47 66 76 95 19 38 96 0A 1A
29 2A 39 3A 48 49 4A 57 58 59 5A 67 68 69 6A 77
78 79 7A 87 88 89 8A 97 98 99 9A A5 A6 A7 A8 A9
AA BB B7 B8 B9 BA C5 C6 C7 C8 C9 CA D4 D5 D6 D7
D8 D9 DA E3 E4 E5 E6 E7 E8 E9 EA F4 F5 F6 F7 F8
F9 FA
Run/Size      Code length  Code word
0/0            3           010
0/1            2           00
0/2            3           011
0/3            3           100

```

0/4	4	1010
0/5	5	11000
0/6	6	111000
0/7	8	11110100
0/8	10	1111110010
0/9	18	1111111110011000
0/A	18	1111111110111011
1/1	4	1011
1/2	5	11001
1/3	7	1110110
1/4	8	11110101
1/5	9	111110010
1/6	11	11111110100
1/7	16	1111111110010101
1/8	16	1111111110100000
1/9	16	1111111110111000
1/A	18	1111111110111100
2/1	5	11010
2/2	7	1110111
2/3	9	111110011
2/4	10	1111110011
2/5	12	111111110100
2/6	16	111111111000111
2/7	16	1111111110100001
2/8	16	1111111110101011
2/9	16	1111111110111101
2/A	18	1111111110111110
3/1	5	11011
3/2	8	11110110
3/3	10	1111110100
3/4	11	11111110101
3/5	12	111111110101
3/6	16	1111111110010110
3/7	16	1111111110100101
3/8	16	1111111110111001
3/9	16	1111111110111111
3/A	18	1111111111000000
4/1	8	111001
4/2	9	111110100
4/3	11	11111110110
4/4	16	1111111110010000
4/5	16	1111111110011001
4/6	16	1111111110100110
4/7	16	1111111110110100
4/8	16	1111111111000001
4/9	16	1111111111000010
4/A	16	1111111111000011
5/1	6	111010
5/2	9	111110101
5/3	11	11111110111
5/4	16	1111111110010001
5/5	16	1111111110100010
5/6	18	1111111110101111
5/7	16	1111111111000100
5/8	16	1111111111000101
5/9	16	1111111111000110
5/A	18	1111111111000111
6/1	7	1111000
6/2	10	11111110101
6/3	12	111111110110
6/4	16	1111111110011010
6/5	18	1111111110100111
6/6	18	1111111110110101
6/7	16	1111111111001000
6/8	18	1111111111001001
6/9	18	1111111111001010

6/A	16	1111111111001011
7/1	7	1111001
7/2	10	1111110110
7/3	12	111111110111
7/4	16	1111111110011011
7/5	16	1111111110110000
7/6	16	1111111110110110
7/7	16	1111111111001100
7/8	16	1111111111001101
7/9	16	1111111111001110
7/A	16	1111111111001111
8/1	8	11110111
8/2	11	11111111000
8/3	16	1111111110010010
8/4	16	1111111110011111
8/5	16	1111111110101100
8/6	16	1111111110110001
8/7	16	1111111111010000
8/8	16	1111111111010001
8/9	16	1111111111010010
8/A	16	1111111111010011
9/1	8	11111000
9/2	11	11111111001
9/3	16	1111111110010011
9/4	16	1111111110011100
9/5	16	1111111110110111
9/6	16	1111111110111010
9/7	16	1111111111010100
9/8	16	1111111111010101
9/9	16	1111111111010110
9/A	16	1111111111010111
A/1	9	111110110
A/2	13	1111111110000
A/3	16	1111111110011101
A/4	16	1111111110101000
A/5	16	1111111111011000
A/6	16	1111111111011001
A/7	16	1111111111011010
A/8	16	1111111111011011
A/9	16	1111111111011100
A/A	16	1111111111011101
B/1	9	111110111
B/2	14	11111111100010
B/3	16	1111111110010111
B/4	16	1111111110101001
B/5	16	1111111110110010
B/6	16	1111111111011110
B/7	16	1111111111011111
B/8	16	1111111111100000
B/9	16	1111111111100001
B/A	16	1111111111100010
C/1	9	111111000
C/2	16	1111111110010100
C/3	16	1111111110101010
C/4	16	1111111110101101
C/5	16	111111111100011
C/6	16	1111111111100100
C/7	16	1111111111100101
C/8	16	1111111111100110
C/9	16	1111111111100111
C/A	16	1111111111101000
D/1	10	1111110111
D/2	16	1111111110011110
D/3	16	1111111110110011
D/4	16	1111111111101001
D/5	16	1111111111101010

D/6	16	111111111101011
D/7	16	111111111101100
D/8	16	111111111101101
D/9	16	111111111101110
D/A	16	111111111101111
E/1	10	111111000
E/2	16	111111110100011
E/3	16	111111111110000
E/4	16	111111111110001
E/5	16	111111111110010
E/6	16	111111111110011
E/7	16	111111111110100
E/8	16	111111111110101
E/9	16	111111111110110
E/A	16	111111111110111
F/0	10	111111001
F/1	15	11111111000110
F/2	16	111111110100100
F/3	16	111111110101110
F/4	16	111111111111000
F/5	16	111111111111001
F/6	16	111111111111010
F/7	16	111111111111011
F/8	16	111111111111100
F/9	16	111111111111101
F/A	16	111111111111110

\*\*\* a\*, b\* AC Huffman table \*\*\*

\*\* BITS in hex. \*\*

00	02	02	01	03	02	03	05	05	07	01	01	01	02	00	7F
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

\*\* HUFFVAL in hex. \*\*

00	01	02	11	03	12	21	31	04	41	22	51	61	05	13	32
71	F0	81	91	A1	B1	C1	08	14	23	33	42	D1	E1	52	15
43	62	72	F1	07	24	34	82	92	B2	53	A2	C2	16	25	83
A3	D2	08	35	44	63	73	93	B3	E2	17	54	64	84	94	A4
B4	C3	26	45	38	F2	55	74	D3	C4	09	0A	18	19	1A	27
28	29	2A	37	38	39	3A	46	47	48	49	4A	56	57	58	59
5A	65	66	67	68	69	6A	75	76	77	78	79	7A	85	86	87
88	89	8A	95	95	97	98	99	9A	A5	A6	A7	A8	A9	AA	B5
B6	B7	B8	B9	BA	C5	C6	C7	C8	C9	CA	D4	D5	D6	D7	D8
D9	DA	E3	E4	E5	E6	E7	E8	E9	EA	F3	F4	F5	F6	F7	F8
F9	FA														

Run/Size	Code length	Code word
0/0	2	00
0/1	2	01
0/2	3	100
0/3	4	1100
0/4	6	111010
0/5	8	11110110
0/6	10	1111110110
0/7	16	111111110000001
0/8	16	111111110001111
0/9	16	111111110100111
0/A	16	111111110101000
1/1	3	101
1/2	5	11010
1/3	8	11110111
1/4	10	1111110111
1/5	12	111111110110
1/6	16	111111110001010
1/7	16	111111110010111
1/8	16	111111110101001
1/9	16	111111110101010
1/A	16	111111110101011

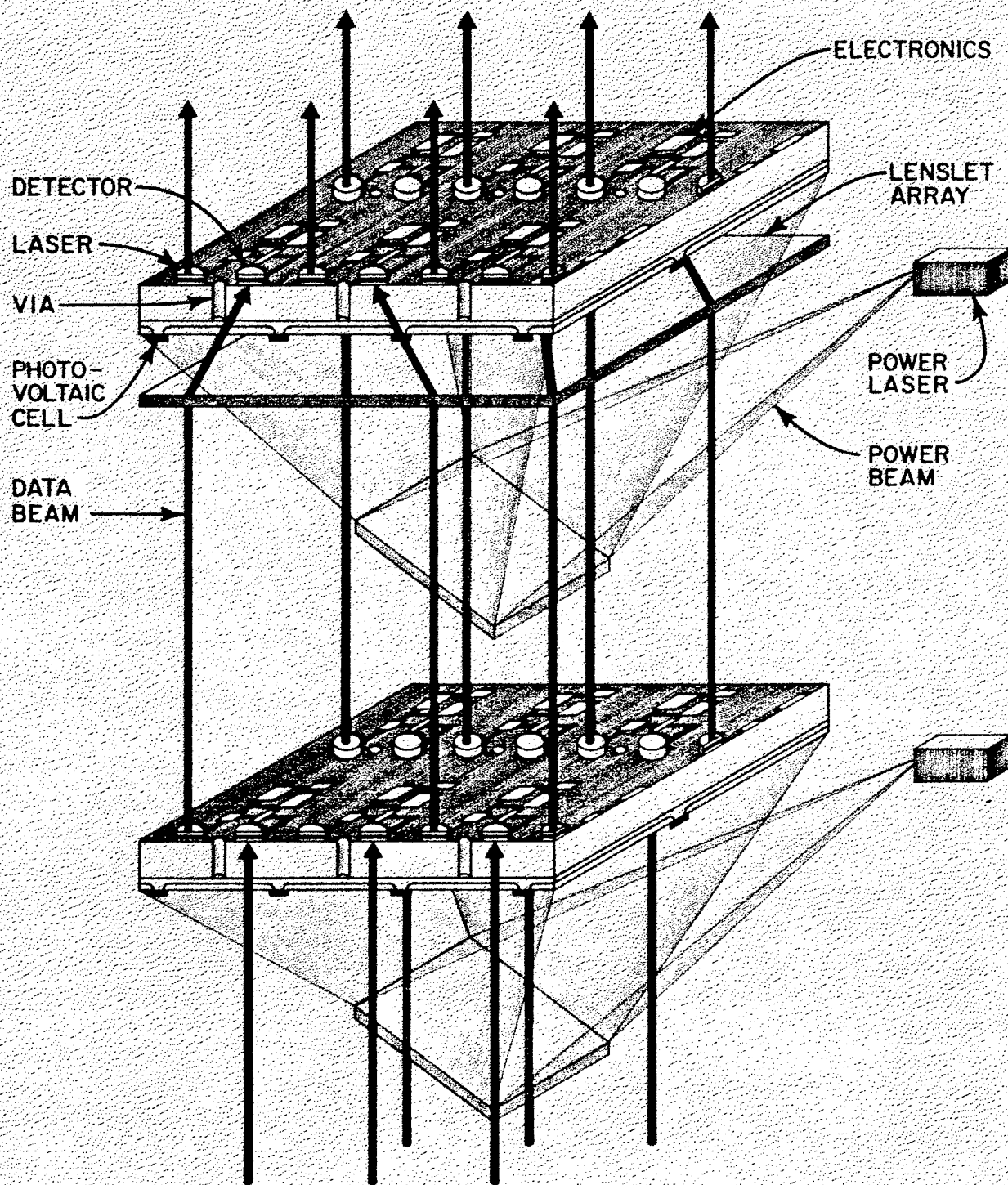
2/1	5	11011
2/2	7	1111000
2/3	10	111111000
2/4	18	111111110000010
2/5	16	111111110001011
2/6	16	111111110011111
2/7	16	111111110101100
2/8	18	111111110101101
2/9	16	111111110101110
2/A	18	111111110101111
3/1	5	11100
3/2	8	11111000
3/3	10	111111001
3/4	18	111111110000011
3/5	16	111111110010000
3/6	16	111111110100001
3/7	16	111111110110000
3/8	16	111111110110001
3/9	18	111111110110010
3/A	18	111111110110011
4/1	8	111011
4/2	10	1111111010
4/3	13	1111111101110
4/4	18	111111110010001
4/5	16	111111110100000
4/6	16	111111110110100
4/7	18	111111110110101
4/8	16	111111110110110
4/9	16	111111110110111
4/A	16	111111110111000
5/1	7	1111001
5/2	11	11111111010
5/3	16	111111110000111
5/4	16	111111110011000
5/5	18	111111110100011
5/6	16	111111110111001
5/7	16	111111110111010
5/8	16	111111110111011
5/9	16	111111110111100
5/A	16	111111110111101
6/1	7	1111010
6/2	14	11111111011110
6/3	16	111111110010010
6/4	18	111111110011001
6/5	18	111111110111110
6/6	16	111111110111111
6/7	16	111111111000000
6/8	18	111111111000001
6/9	16	111111111000010
6/A	16	111111111000011
7/1	8	11111001
7/2	14	11111111011111
7/3	16	111111110010011
7/4	16	111111110100100
7/5	16	111111111000100
7/6	16	111111111000101
7/7	16	111111111000110
7/8	16	111111111000111
7/9	16	111111111001000
7/A	16	111111111001001
8/1	9	111110110
8/2	18	1111111110000100
8/3	18	1111111110001100
8/4	18	1111111110011010
8/5	18	111111111001010
8/6	18	111111111001011



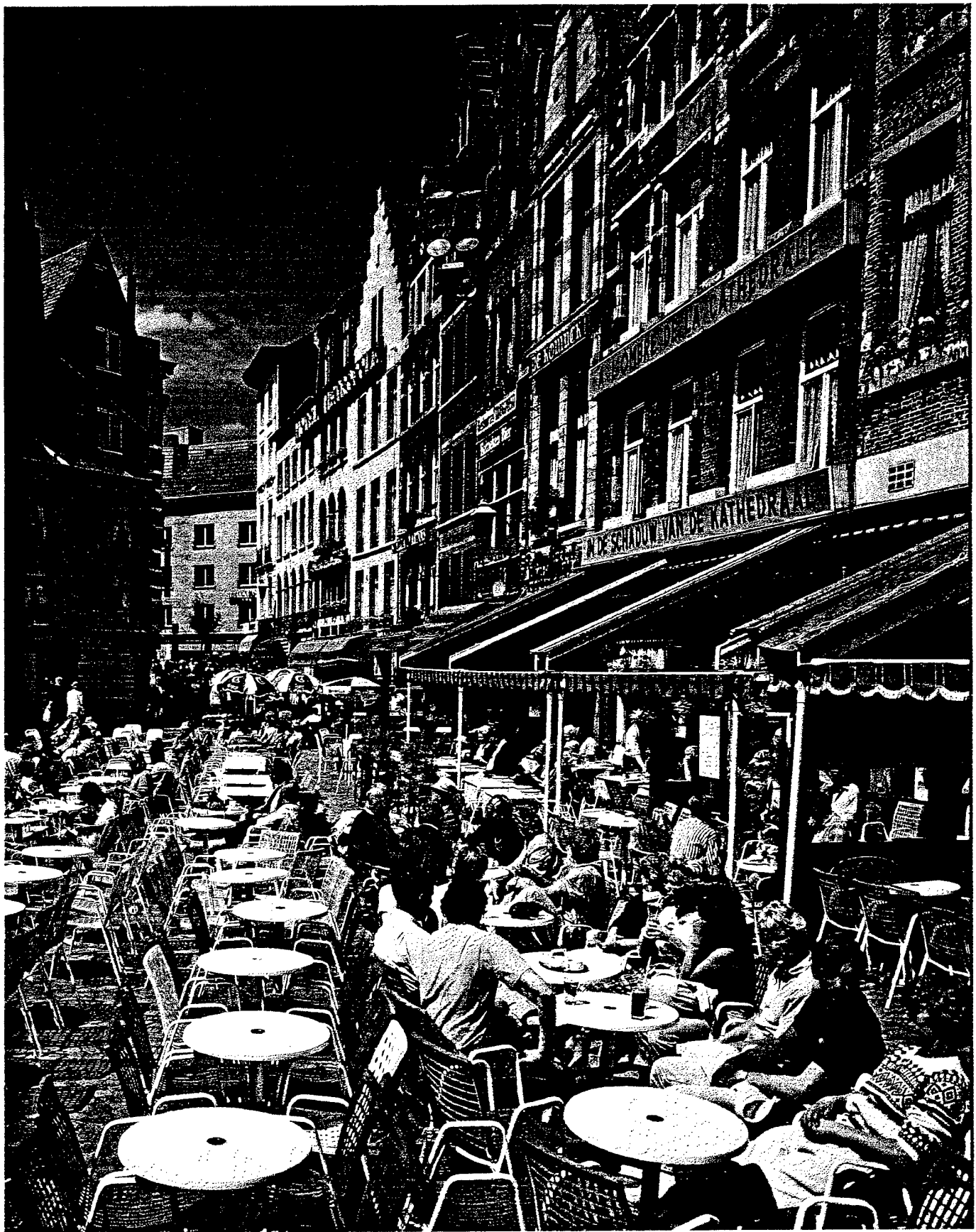
8/7	16	111111111001100
8/8	16	111111111001101
8/9	16	111111111001110
8/A	16	111111111001111
9/1	9	111110111
9/2	18	1111111110000101
9/3	16	1111111110010100
9/4	18	1111111110011011
9/5	16	1111111111010000
9/6	16	1111111111010001
9/7	16	1111111111010010
9/8	16	1111111111010011
9/9	16	1111111111010100
9/A	16	1111111111010101
A/1	9	111111000
A/2	16	1111111110001000
A/3	16	1111111110001101
A/4	16	1111111110011100
A/5	16	1111111111010110
A/6	16	1111111111010111
A/7	16	1111111111011000
A/8	16	1111111111011001
A/9	16	1111111111011010
A/A	16	1111111111011011
B/1	9	111111001
B/2	16	1111111110000110
B/3	16	1111111110010101
B/4	16	1111111110011101
B/5	16	1111111111011100
B/6	16	1111111111011101
B/7	16	1111111111011110
B/8	16	1111111111011111
B/9	16	1111111111000000
B/A	16	1111111111000001
C/1	9	111111010
C/2	16	1111111110001001
C/3	16	1111111110011110
C/4	16	11111111110100110
C/5	16	1111111111100010
C/6	16	1111111111100011
C/7	16	1111111111100100
C/8	16	1111111111100101
C/9	16	1111111111100110
C/A	16	1111111111100111
D/1	10	1111111011
D/2	16	1111111110001110
D/3	16	11111111110100101
D/4	16	1111111111101000
D/5	16	1111111111101001
D/6	16	1111111111101010
D/7	16	1111111111101011
D/8	16	1111111111101100
D/9	16	1111111111101101
D/A	16	1111111111101110
E/1	10	1111111100
E/2	18	1111111110010110
E/3	16	1111111111101111
E/4	16	1111111111100000
E/5	16	1111111111100001
E/6	16	1111111111100010
E/7	16	1111111111100011
E/8	16	1111111111101000
E/9	16	1111111111101001
E/A	16	1111111111101010
F/0	8	11111010
F/1	18	1111111110000000

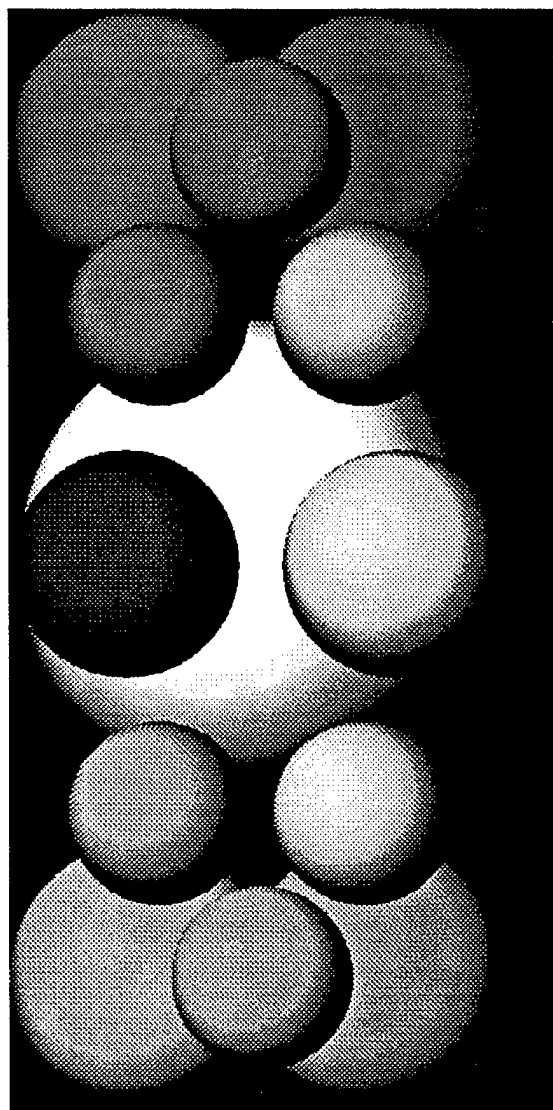
F/2	16	1111111110100010
F/3	16	1111111111110111
F/4	16	1111111111111000
F/5	16	1111111111111001
F/6	16	1111111111111010
F/7	16	1111111111111011
F/8	16	1111111111111100
F/9	16	1111111111111101
F/A	16	1111111111111110

# INFORMATION PROCESSING



APPOT LAB





FAX BALLS



Georges de La Tour (1593-1652) : "Le Triomphe de la Vierge"  
Digitized by LRMF and ENST - Distributed only for €

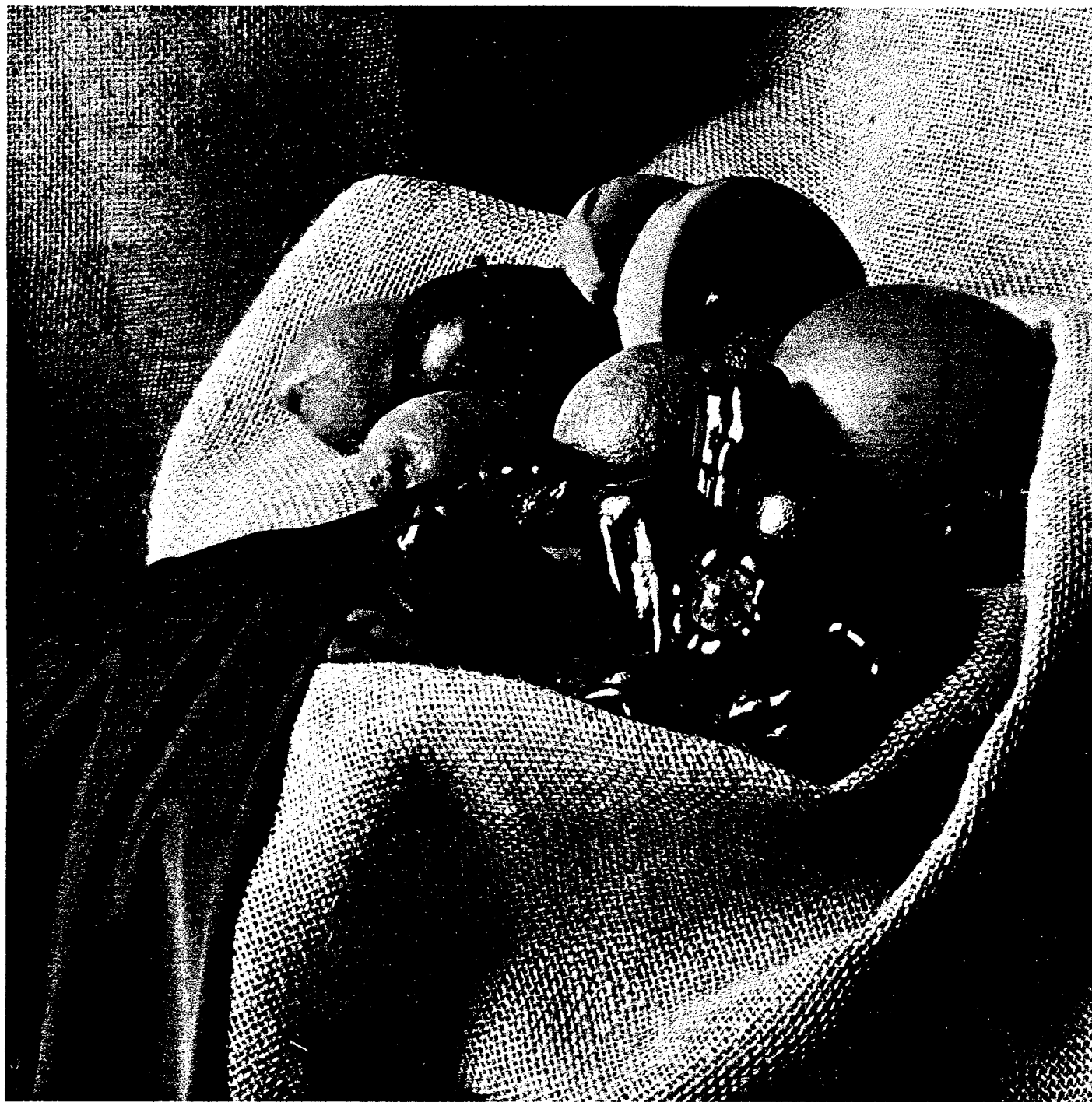


*cheur", Musée du Louvre, Paris, France*  
*CITT color facsimile tests - Not for commercial use*



PM003\_NL





SA001\_NL



TOYS LAB